ARMY RESEARCH LABORATORY

Antenna Transient Compensation

N. Tesny, M. Litz, L. Dilks, and D. Conrad

ARL-TR-2229

July 2001

Approved for public release; distribution unlimited.

Army Research Laboratory

Adelphi, MD 20783-1197

ARL-TR-2229 **July 2001**

Antenna Transient Compensation

N. Tesny, M. Litz, L. Dilks, and D. Conrad

Sensors and Electron Devices Directorate

Abstract

An automated method has been implemented in MATLAB® to compensate for signal dispersion in antenna structures. We have explored postprocessing techniques that involve frequency transforms and deconvolution. The method has been applied to transient signals measured from a variety of different antennas and impulse sources. The technique has proved to be a valuable tool in reconstructing fast transient signals with inexpensive high-gain log-periodic antennas instead of more expensive, high fidelity wideband horns.

Contents

Appendices

Figures

Tables

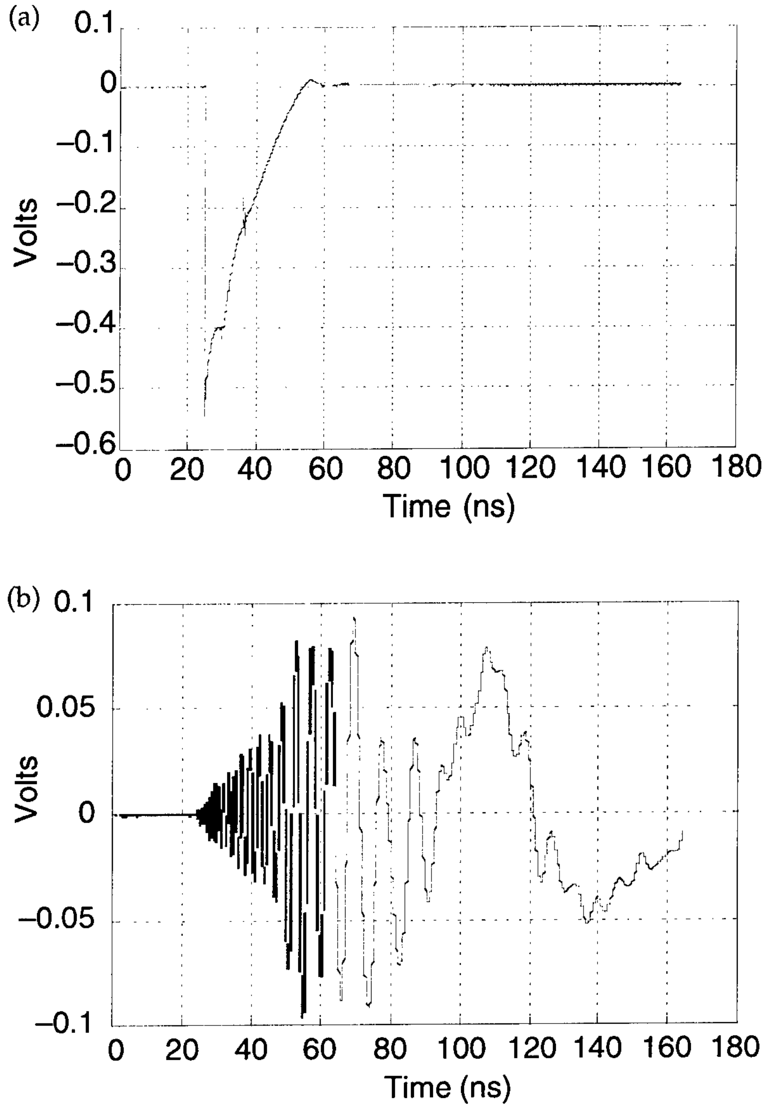
1. Background

High fidelity wideband antennas have been developed to preserve the pulse shape of ultra wideband transient waveforms [1]. These antennas are typically horn structures that occupy large volumes with low gain. The gain is lower for the longer wavelengths because the gain must vary linearly with frequency to maintain a constant effective length. These antennas have been very effective for large signal strength, high fidelity data-collection requirements.

Commercially available, high-gain log-periodic (LP) antennas would be useful for capturing small signal transients that are below the minimum detectable signal level of the bulkier large aperture antennas. The higher gain LP antennas act as filters that modify the phase and amplitude of the input signal as a function of frequency. Every antenna disperses a pulsed signal to some extent. Some antennas are more dispersive than others because of materials used, construction, or both. The purpose of this work is to correct for the dispersion with digital filter postprocessing and to reconstruct the original signal.

Dispersion from an antenna is characterized by group delay and amplitude distortion. Every antenna has a characteristic dispersion. Some antennas are large physical structures that delay the signal because they have many different length-resonant elements (as in a log-periodic antenna); others introduce signal dispersion because cable characteristics and/or baluns are added to the channel path. Figure 1 shows the dispersion of a pulse caused by a log-periodic antenna.

Figure 1. (a) The direct output of a Picosecond Pulse Labs 4015C pulse generator and (b) as transmitted between a pair of log-periodic antennas.



2. Approach

The approach is to measure the response of a pair (transmitting/receiving) of identical antennas to a wideband source. An impulse source generates a signal that is transmitted through a radiating antenna placed at one end of an anechoic chamber. To obtain the true response of the antenna, a very wideband (0 to 10 GHz) pulser is used. A receiving antenna is connected to a digital oscilloscope at the other end of the anechoic chamber. The digital oscilloscope acts as a base-band receiver. Data acquired by the digital oscilloscope are downloaded to a portable computer, where they are filtered and transformed via digital signal processing (DSP) in MATLAB® **[2]** as described next. In addition, the source pulse is entered directly into the digital oscilloscope and recorded. These two signals are deconvolved to obtain the antenna response in the following manner:

$$S_{\mathrm{TransFunc,FFT}} = \frac{\mathrm{FFT}\left(\frac{d}{dt} S_{\mathrm{Direct}}\right)}{\mathrm{FFT}\left(S_{\mathrm{Thru}}\right)} , \qquad (1)$$

in which

S_{Direct} is the direct voltage signal from the pulser,
S_{Thru} is the response of the pulser through the antennas of interest,
$S_{\mathrm{TransFunc,FFT}}$ is the frequency response of the transfer function, and
FFT is the fast Fourier transform.

The transfer function response filter is calculated from the ratio of the direct and transmitted signal. It is then applied to the data taken with the original or other pulsed sources. By applying the transfer function to the measured data, we can remove the dispersive effect of the antennas. This is done in the following manner:

$$S_{\mathrm{Corrected,FFT}} = \frac{\mathrm{FFT}\left(S_{\mathrm{Measured}}\right)}{\left(S_{\mathrm{TransFunc,FFT}}\right)} \qquad (2)$$

$$S_{\mathrm{Corrected,FFT}} = \mathrm{I}_{\mathrm{FFT}}\left(S_{\mathrm{Corrected,FFT}}\right) \qquad (3)$$

in which

S_{Measured} is the measured signal from the receiving antenna,
$S_{\mathrm{Corrected}}$ is the corrected time response of the pulser through the antennas of interest,
$S_{\mathrm{TransFunc,FFT}}$ is the frequency response of the transfer function, and
$\mathrm{I}_{\mathrm{FFT}}$ is the inverse fast Fourier transform (FFT).

3. Experimental Configuration

Figure 2 shows the experimental arrangement. The configuration consists of a source for generating fast pulses connected to a transmitting antenna and a Tektronix 11801C sampling oscilloscope connected to a receiving antenna. The 11801C sampling oscilloscope uses an SD-24 sampling head that is capable of measurements as great as 20 GHz. The sampling oscilloscope is connected to the source via a cable for triggering.

Several impulse sources have been used during the investigations to generate the wideband signals. These include (1) the PicoSecond Pulse Labs (PSPL) 4015C pulser, which has a nominal rise time of 17 ps and peak output of 9 V; (2) the Avtech pulser, which supports variable rise time and pulse width; and (3) the hand-held (HH2) generator, which is a reed switch pulser built in the Amy Research Laboratory for impulse applications [3]. These sources are listed in table 1.

Antenna pairs that were evaluated and compared to each other include a commercial log-periodic antenna, an EMCO double-ridge wave guide horn, and two versions of a resistively loaded transverse electromagnetic horn labeled as T2 and T3. The characteristics of these antennas were measured with identical pairs of antennas during these experiments and are listed in table 2.

Figure 2. Experimental schematic.

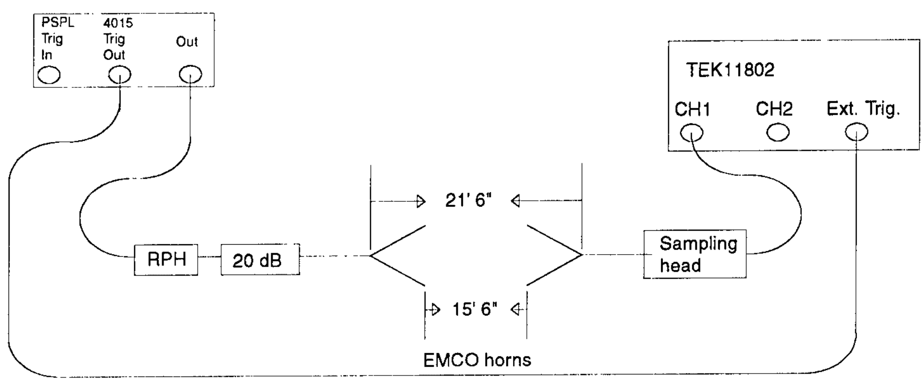


Table 1. Sources used.

Source	Rise time	Output (V)	Repetition rate
PicoSecond Pulse Labs 4015C	17 ps, nominal	9	10 kHz
Hand-held pulser (HH2)	100 ps	700	100 Hz
Avtech pulser	5 to 100 ns	300	1 kHz

Table 2. Antennas used.

Antenna	Bandwidth	Gain (dB)
T3	20 MHz to 8 GHz	–2
T2	100 MHz to 4 GHz	–5
Log Periodic	100 MHz to 1.3 GHz	8
EMCO	200 MHz to 2 GHz	12

4. Procedure

The transfer function technique described in section 2 was applied to two different pulsed sources and four different antenna pairs. As mentioned earlier, two waveforms are required: (1) the signal directly from the impulse source, and (2) the signal from the source through the antenna pair. The signal's input to and output from the T3 antenna pair (for example) are shown in figure 3. The derivative of the "direct" pulse is taken. This is done because an antenna transmits according to the derivative of the signal fed into it. Moving charges are required to generate the crossed fields in the antenna. A DC signal will not radiate. Thus, the derivative represents the ideal behavior of radiation generated in an antenna. However, since no antenna has an infinite bandwidth, signals through them will be limited in bandwidth, as shown in figure 3.

The FFT is calculated for the two waveforms (direct and through antennas) shown in figure 4. We then obtain the transfer by dividing the two FFT signals, as described in equation (1). This transfer function (see fig. 5), which is in the frequency domain, is then plotted and saved.

Figure 6 shows the complete flow chart for this process. This flow chart has been converted to a MATLAB [2] script file. The code is shown in appendix A. Measurements of "direct" and "through" are saved. These files are input to the code through a series of menus. Any attenuation or amplification used during the measurements is also input to obtain proper scaling of the waveforms. After the user enters these data, the transfer function is computed and saved in ASCII format for later use in the correction process.

This transfer function is then used as a filter and applied to signals received through the antennas from other sources. Specifically, the signal transmitted through the antenna pair, which will be referred to as the "thru" signal, is read. An FFT is taken of the "thru" signal. The transfer function is also read and interpolated, if necessary, to match the sample interval and length of the "thru" waveform. The two signals are then divided as described in equation (2) to create a corrected frequency-domain file. This file is then filtered with a band-pass filter to eliminate high and low frequency noise and anomalies that are outside the receiving antenna's operating band. These anomalies are spuriously generated during the compensation process. Then, an inverse fast Fourier transform is applied to obtain the corrected time-domain signal. It is then plotted and saved. Figure 7 shows the flow chart for this process. The code that performs these operations is also shown in appendix A. The program uses a series of menus to guide the user through the entire process.

Figure 3. Signals input to (green) and output from (blue) an antenna pair.

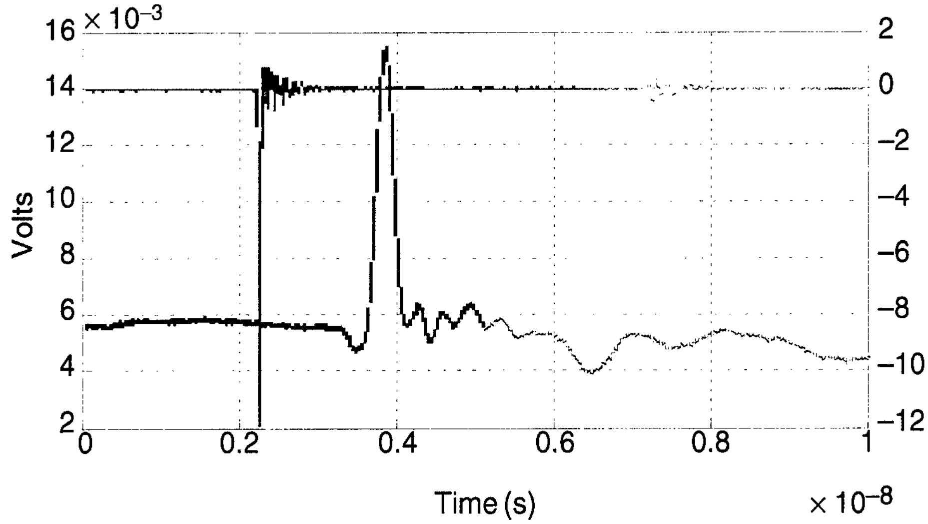


Figure 4. Fast Fourier transforms (FFTs) of the input and output signals of figure 3.

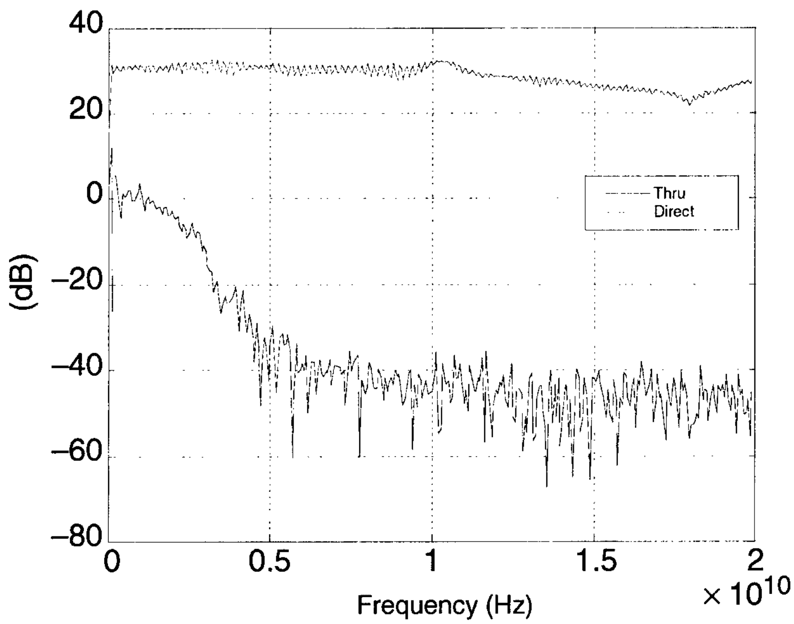


Figure 5. Transfer function generated from input and output waveforms.

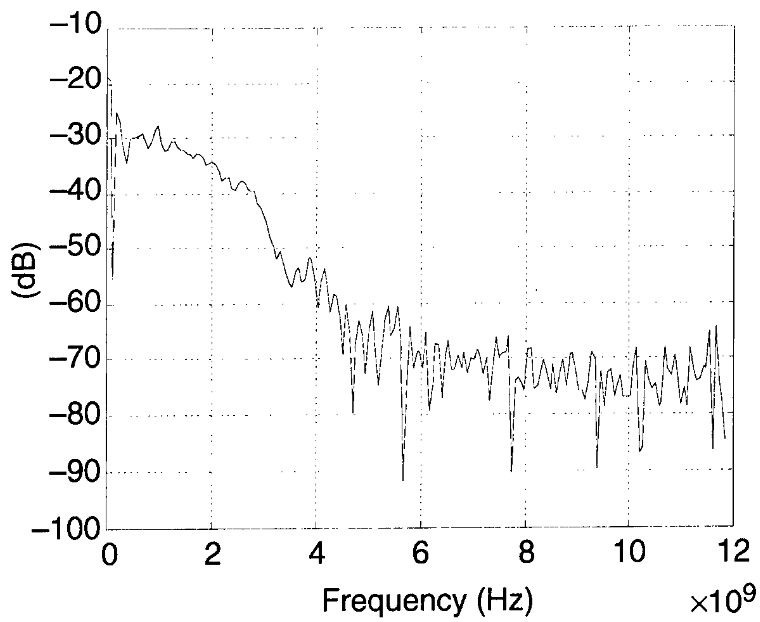


Figure 6. Flowchart of process to generate transfer function.

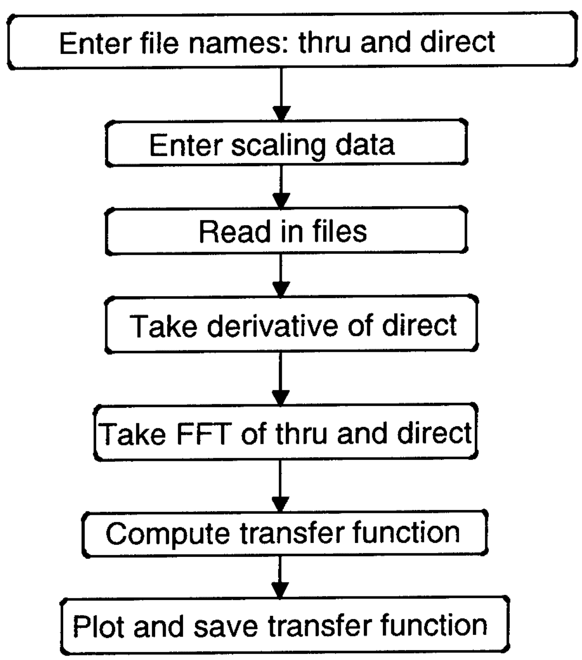
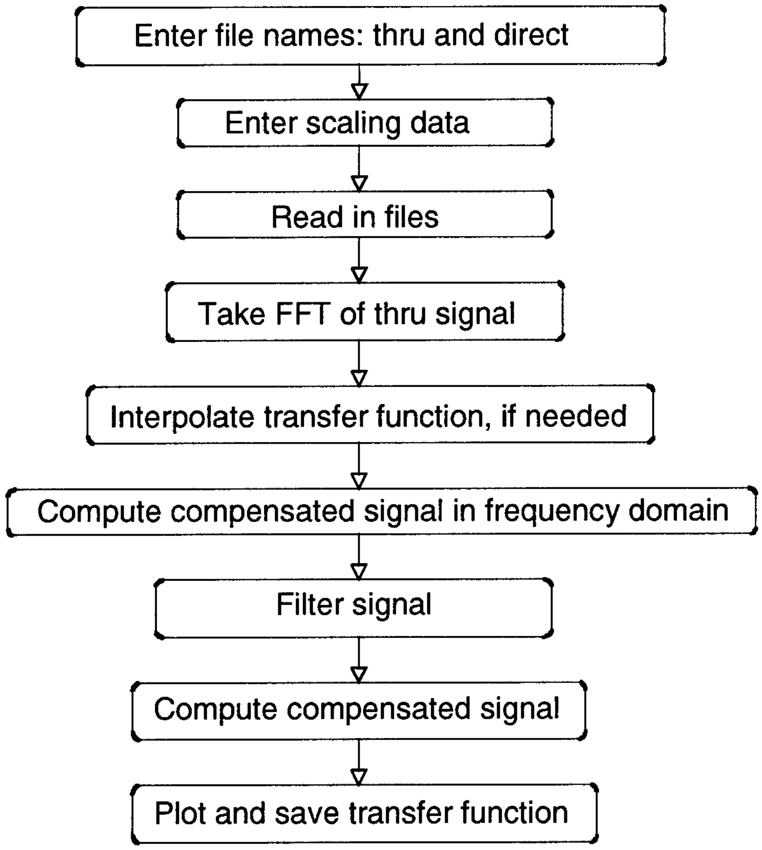


Figure 7. Flowchart of signal correction process.



5. Results

The effectiveness of the compensation algorithm varied for the multiple antenna-source combinations. Figure 8 shows relatively good reconstruction when the hand-held source and T2 antenna pair were used. The first plot in each figure shows (1) signal through the antenna pair (blue), (2) compensated version of this signal (green), and (3) derivative of the signal taken directly from the output of the pulser (red). The rise times of these signals are included in the legend for comparison. The second plot shows the frequency responses (FFTs) of these signals. Overall effectiveness was based on how close the reconstructed or compensated signal was to the derivative of the direct signal. Factors used to compare the signals included the rise times, pulse widths, and pulse shapes in the time domain, and the frequency content and amplitudes in the frequency domain. As the sample rates (and Δt) vary from the examples shown, so does the calculated derivation of the input voltage. This variation is a natural result of finite differences inherent in DSP.

Figure 9 shows the reconstruction when the hand-held source was used with the EMCO antennas. Based on the rise times, pulse widths, and pulse shapes of the compensated and original signals, very good signal reconstruction was achieved in this case.

Figures 10 and 11 show lower quality reconstruction of the received signal. One of the main reasons for the poor compensation was that our enclosed test chamber prevented adequate propagation of signals below 200 MHz. At 200 MHz, a significant ringing was apparent in some of the signals, which was also attributable to the use of an enclosed metal chamber. We believe this is why the results were poor for the Avtech pulser, which consists primarily of signals below 200 MHz. Results with other source-antenna combinations are shown in appendix B.

The width of pulses produced by the Avtech source varied from pulse to pulse. As a result, the oscilloscope's measuring method of sampling produced a pulse with a distorted trailing edge. This resulted in erratic frequency transforms because of jitter in the trailing edge of the pulse, which is shown in figure 12.

The compensatory method used signals that were collected with different sampling rates and recording lengths. The effectiveness of these multirate compensations varied throughout the combinations of antennas, sources, and recording devices. The complete data set and results of its corrections are shown in appendix B.

Figure 8.
Reconstruction of signal from HH2 source through T2 antenna pair (see text for discussion).

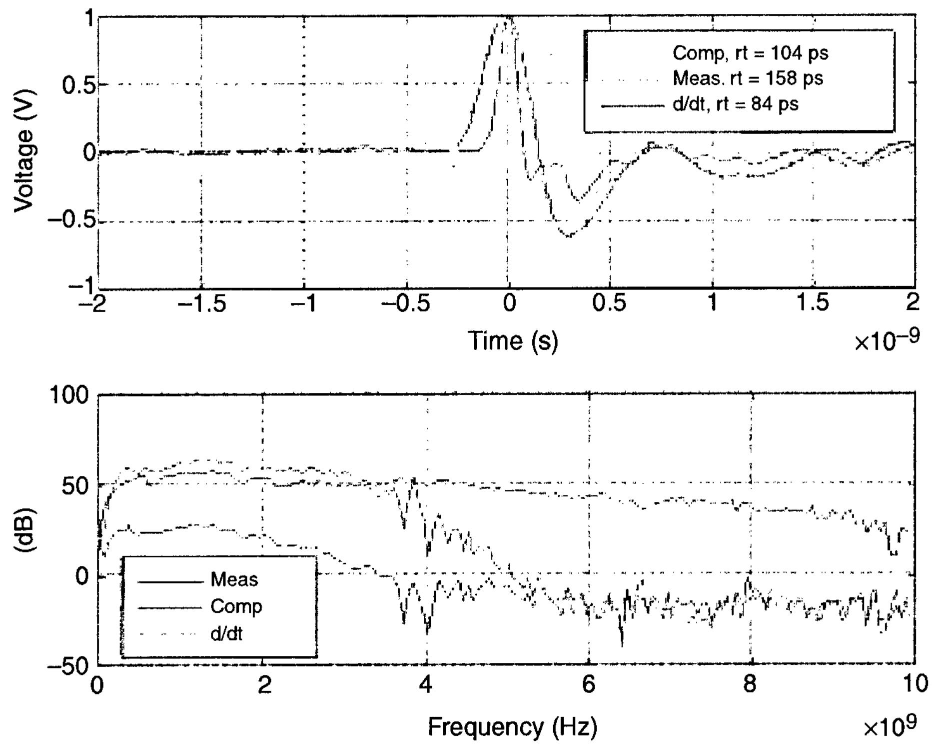


Figure 9.
Reconstruction of signal from HH2 source through EMCO antenna pair.

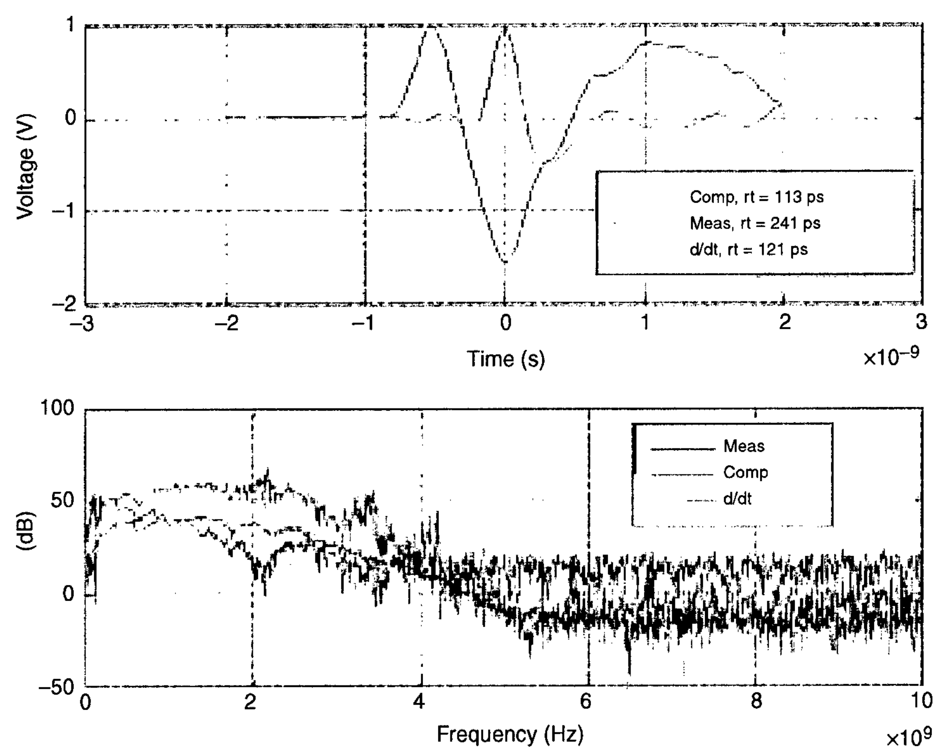


Figure 10.
Reconstruction of signal from Avtech source through T2 antenna pair.

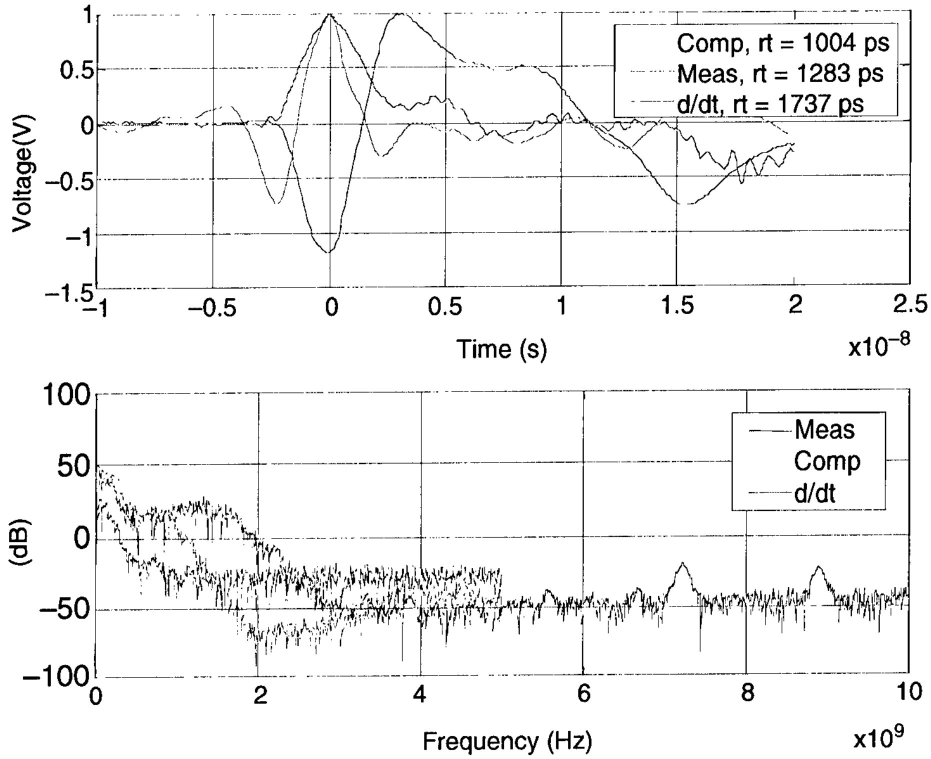


Figure 11.
Reconstruction of signal from Avtech source through EMCO antenna pair.

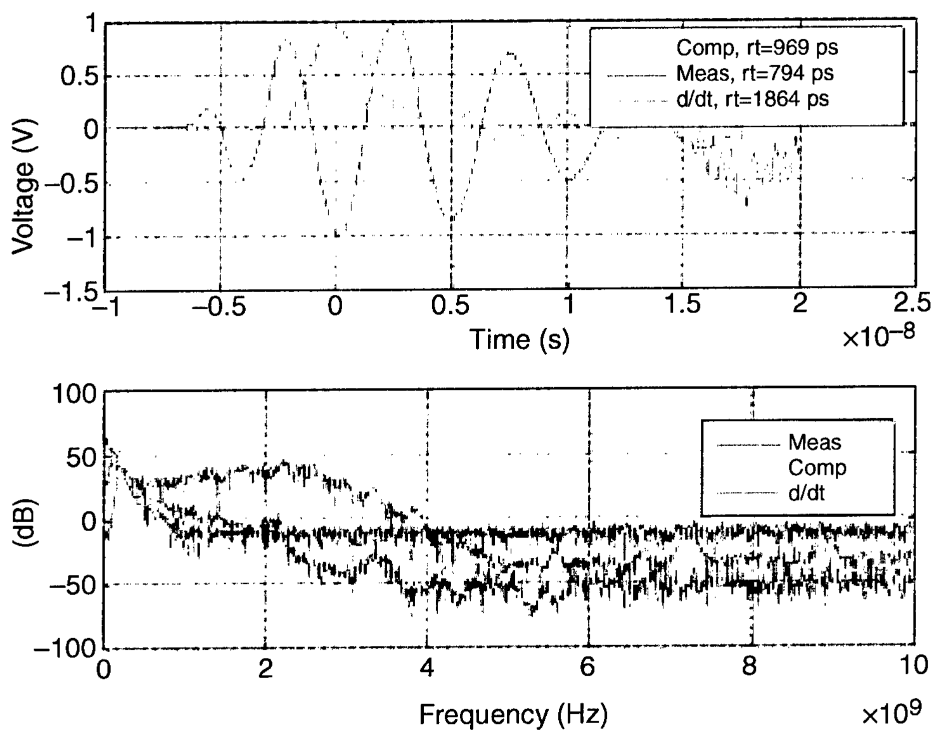
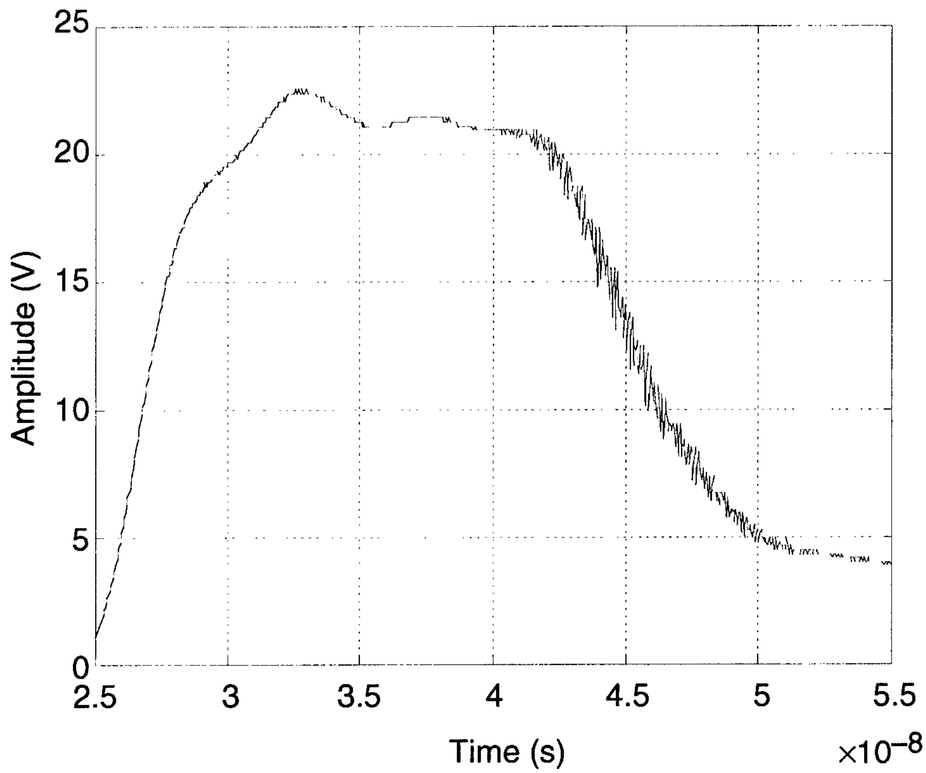


Figure 12. Jitter in Avtech source.



6. Conclusions

Wideband sources and antennas provide better compensation. Care must be taken to ensure that the entire bandwidth range is covered when one is obtaining the antenna response, that is, the reference measurement, especially for the lower frequencies. Our instruments limited us to 1,024 samples, which may not have been adequate to calculate the low frequency response. With the latest versions of fast analog-to-ditigal converters and megabyte record lengths available, this problem should be eliminated.

Future work being planned will include correcting for measurements of antennas that are off boresight, correcting for measurements performed outside the normal operating frequency ranges, using full discrete Fourier transforms instead of FFTs for signal correction, and using other digital filtering in the correction process.

Acknowledgments

We would like to thank Dave Conrad and Derwin Washington, of the Sensors and Electronic Devices Directorate, who were crucial in performing the experiments.

References

[1] Morgan, M. A., and R. C. Robertson, *Ultra-Wide-Band Impulse Antenna Study and Prototype Design*, U.S. Navy Naval Postgraduate School, Monterey, CA, final report to U.S. Army CECOM (12 April 1993).

[2] MATLAB V5.3, The Mathworks Inc. (January 1999).

[3] Litz, M. S., D. C. Judy, D. M. Weidenheimer, and B. Jenkins, *Compact Impulse Source for Wideband Signal Calibrations and General Laboratory Use*, U.S. Army Research Laboratory, ARL-TR-2117 (April 2000).

Appendix A. Results

This appendix includes the internally documented script files. The MATLAB [2] code developed for these calculations is shown below.

```
% comp_tranFuncGen.m
% 4/99 N Tesny
% generates FFT transfer function of two pulser data signals
% this program doesn't handle differing sample rates yet

FALSE=0;TRUE=~FALSE;
%filpre = 'g:/uwb/other_subjects/pspl/'
filpre = 'g:\uwb\other_subjects\lpDecon\';

% ask user name of file for thru:
pname=filpre;
[filthru,pname] = uigetfile([pname,'*.asc'],'Select THRU
file')
if ((filthru(1)~=0) & (filthru(length(filthru))~=0)),
   path_thru=pname;
else
   path_thru=filpre;
end

% enter attenuation for THRU file:
OKflag=FALSE;
while OKflag==FALSE,
    prompt  = {'Enter dB attenuation:','Enter ratio attenua-
tion:'};
    title1   = 'Input Attenuation for THRU file';
    lines= 1;
    def     = {'20','1'};
    answer  = inputdlg(prompt,title1,lines,def)

    f=cell2struct(answer(1),'d',1);
    a1s=f.d;
    a1=str2num(a1s);
    f=cell2struct(answer(2),'d',1);
    a2s=f.d;
    a2=str2num(a2s);

    OKflag=TRUE;
    if (isempty(a1)),
       h=errordlg([a1s,' is not a number'],'Bum Data En-
try');
       OKflag=FALSE;
       waitfor(h);
    elseif (isempty(a2)),
       h=errordlg([a2s,' is not a number'],'Bum Data En-
try');
       OKflag=FALSE;
       waitfor(h);
    end
 end
```

```
 if a2==0,a2=1;end;
 thru_attenuation=a1+20*log10(a2)


% ask user name of file for direct:
pname=filpre;
[fildirect,pname] = uigetfile([pname,'*.asc'],'Select DIRECT
file')
if ((filthru(1)~=0) & (filthru(length(filthru))~=0)),
   path_direct=pname;
else
   path_direct=filpre;
end

% enter attenuation for DIRECT file:
OKflag=FALSE;
while OKflag==FALSE,
    prompt  = {'Enter dB attenuation:','Enter ratio attenua-
tion:'};
    title1   = 'Input Attenuation for DIRECT file';
    lines= 1;
    def     = {'20','1'};
    answer  = inputdlg(prompt,title1,lines,def)

    f=cell2struct(answer(1),'d',1);
    a1s=f.d;
    a1=str2num(a1s);
    f=cell2struct(answer(2),'d',1);
    a2s=f.d;
    a2=str2num(a2s);

    OKflag=TRUE;
    if (isempty(a1)),
       h=errordlg([a1s,' is not a number'],'Bum Data En-
try');
       OKflag=FALSE;
       waitfor(h);
    elseif (isempty(a2)),
       h=errordlg([a2s,' is not a number'],'Bum Data En-
try');
       OKflag=FALSE;
       waitfor(h);
    end
 end
 if a2==0,a2=1;end;
 direct_attenuation=a1+20*log10(a2)




% Read in files:

  filname = strcat(filpre,deblank(filthru));
  [x,y] = ascread(filname);
  scalefactor_thru=10^(thru_attenuation/20);
  y=y * (10^(thru_attenuation/20));
  filname = strcat(filpre,deblank(fildirect));
  [xb,yb] = ascread(filname);
```

```
  yb=yb * (10^(direct_attenuation/20));

% Check if sample rates are the same:
if abs((x(2)-xb(2))/x(2)) > 9e-7,
   h=warndlg(['Sampling rates are different.  Press any key
to interpolate time domain waveforms'],'Warning');
    OKflag=FALSE;
    waitfor(h);

    % interpolate time domain waveforms:
    % assume thru is slower rate than direct:

    % pad direct (yb) out ot end of thru (y). Keep sample
rate:
    tstop=x(length(x));
    dt=xb(2)-xb(1);
    k=length(xb);
    while xb(k)+dt<=tstop,
       k=k+1;
       xb(k)=xb(k-1)+dt;
       yb(k)=0;
    end

    % interpolate y out to same # pts of yb (direct):
    method='spline';
    y=interp1(x,y,xb,method);
    x=xb;

    % trunkate to 2^14=16384:
    x=x(1:2^14);
    xb=xb(1:2^14);
    y=y(1:2^14);
    yb=yb(1:2^14);

end

disp('Interpolation done');



 % take derivative of direct pulse:
  dt=xb(2)-xb(1);
  for i=2:length(yb),
    dy(i)=(yb(i)-yb(i-1)); % dont divide by /dt;
  end
  dy(1)=0;
  [x,y]=baseline2(x,y,10,138); % do zero adjust

  % scale direct deriv:
  y_dir_deriv = dy * max(abs(yb)) / max(abs(dy));

  fa=fft(y);
  fb=fft(y_dir_deriv);
  fc=fa./fb;
  yc=ifft(fc);
  npts2=arraylen(fc);
  b=fix(npts2/2);
  npts=b+1;
```

```
% do a smooth-----------

fc_smooth=fc; % er, dont do a smooth
%[fc_smooth]=smooth_complex(fc,fix(.005*length(fc)));

[I,J]=size(x);
a=x(2)-x(1);
fx=(0:npts-1)/((J*a));
fx2=(0:npts2-1)/((J*a));

fbD=fft(yb); % take fft of direct

% plot out direct & fft, direct-deriv & fft:

Idir=1;
displayfreq=11.9e9;
displayfreq=min(displayfreq,fx(length(fx)));
while fx(Idir)<displayfreq,    Idir=Idir+1;  end
Idir=Idir-1;

[pk,fw,ris1,xjunk,yjunk] = stats(x,abs(y));
[pk,fw,ris2,xjunk,yjunk] = stats(xb,abs(yb));
figure,suptitle (['Direct file:',fildirect]);
subplot(2,1,1),plot(xb,yb);grid on;
xlabel('time (S)'); ylabel('Voltage (V)');
%title('Response of PSPL Pulser');
%legend('thru','direct');
v=axis;
a=v(1) + (v(2)-v(1))*.55;
bb = v(3) + ((v(4)-v(3)) * 0.95);
text(a,bb,['rise time: ',num2str(ris1/1e-12),' pS']);
%a=v(1) + (v(2)-v(1))*.55;
%b=v(3) + (v(4)-v(3))*.90;
% text(a,b,['fall time(green): ',num2str(risb/1e-12),'
pS']);

subplot(2,1,2),plot(fx(1:Idir),
20*log10(abs(fbD(1:Idir)))); grid on;
xlabel('frequency (Hz)'); ylabel('(V/Hz)');
%[xf,yf]=fft_ps(x,y);
%[xbf,ybf]=fft_ps(xb,yb);
%xcf=xf;

figure,suptitle (['Direct-deriv :',fildirect]);
subplot(2,1,1),plot(xb,y_dir_deriv);grid on;
xlabel('time (S)'); ylabel('normalized (V/S)');
%title('Response of PSPL Pulser');
%legend('thru','direct');
v=axis;
a=v(1) + (v(2)-v(1))*.55;
bb=v(3) + (v(4)-v(3))*.95;
text(a,bb,['rise time: ',num2str(ris2/1e-12),' pS']);
%a=v(1) + (v(2)-v(1))*.55;
%b=v(3) + (v(4)-v(3))*.90;
% text(a,b,['fall time(green): ',num2str(risb/1e-12),'
pS']);
```

```
  subplot(2,1,2),plot(fx(1:Idir),
20*log10(abs(fb(1:Idir)))); grid on;
  xlabel('frequency (Hz)'); ylabel('(V/Hz)');
  %[xf,yf]=fft_ps(x,y);
  %[xbf,ybf]=fft_ps(xb,yb);
  %xcf=xf;


  %
  % let user do filtering of the transfer function:----------
---------
  %
  %[x1,y1] =
gin1(fx,20*log10(abs(fc_smooth(1:b+1))),'Hz','dB')

  %y2=10^(y1/20);
  %[xt,yt]=floor_first(fx,abs(fc(1:b+1)),x1,y2);
  %% take the max of y2 and fc(i):
  %for i=1:k-1,
  %  if y2>abs(fc(i))
  %    fc_new(i) = (y2 ./ abs(fc(i))) .* fc(i);
  %  end
  %end
  %% now do back half:
  %for i=m-(k-1) : m,
  %  if y2>abs(fc(i))
  %    fc_new(i) = (y2 ./ abs(fc(i))) .* fc(i);
  %  end
  %end



  [pk,fw,ris,x,yjunk] = stats(x,abs(y));
  [pk,fw,risb,xb,yjunk] = stats(xb,abs(yb));
  figure,suptitle (filthru);
  subplot(2,2,1),plotyy(x,y,xb,yb);grid on;
  xlabel('time (S)'); ylabel('Voltage (V)');
  title('Response of PSPL Pulser');
  %legend('thru','direct');
  %v=axis;
  %a=v(1) + (v(2)-v(1))*.55;
  %b=v(3) + (v(4)-v(3))*.95;
   % text(a,b,['fall time (blue): ',num2str(ris/1e-12),'
pS']);
  %a=v(1) + (v(2)-v(1))*.55;
  %b=v(3) + (v(4)-v(3))*.90;
   % text(a,b,['fall time(green): ',num2str(risb/1e-12),'
pS']);
  %subplot(2,1,1),plot(x,y); grid on;
  %[xf,yf]=fft_ps(x,y);
  %[xbf,ybf]=fft_ps(xb,yb);
  %xcf=xf;


  f2=fa(1:b+1);
  f2(1)=f2(1)/2;
  f3=fb(1:b+1);
  f3(1)=f3(1)/2;
```

```
J=810; % J=200;
subplot(2,2,2),plot(fx(1:J),20*log10(abs(f2(1:J))),fx(1:J),20*log10(abs(f3(1:J))));grid
on;
xlabel('frequency (Hz)'); ylabel('(dB)');
title('FFTs of PSPL data');
legend('thru','direct',0);

%subplot(2,2,3),plot(fx,20*log10(abs(fc(1:b+1))));grid on;
%xlabel('frequency (Hz)'); ylabel('(dB)');
%title('FFT of PSPL data');

% plot smoothed curve:
subplot(2,2,3),plot(fx,20*log10(abs(fc(1:b+1))),fx,20*log10(abs(fc_smooth(1:b+1))))
xlabel('frequency (Hz)'); ylabel('(dB)');grid on;
title('Transfer funct');
legend('original','smoothed',0);

% plot filtered curve but only up to 10 GHz:
I=1;
while fx(I)<10e9,
  I=I+1;
end
I=I-1;
fc_new = fc_smooth;

subplot(2,2,4),plot(fx(1:I),20*log10(abs(fc_smooth(1:I))),fx(1:I),20*log10(abs(fc_new(1:I))))
xlabel('frequency (Hz)'); ylabel('(dB)');grid on;
title('Transfer funct');
legend('smoothed','truncated',0);

f2=fc_new(1:b+1);
f2(1)=f2(1)/2;
figure;plot(fx2,20*log10(abs(fc_new)));grid on;
xlabel('frequency (Hz)'); ylabel('(dB)');
title('Transfer Function of thru using PSPL data');

yc_new=ifft(fc_new);
figure;plot(x,real(yc_new));grid on;
xlabel('time (S)'); ylabel('normalized response ()');
title('Impulse Response of thru using PSPL data');

% Begin the plots that we want:

figure,%title (filthru);
plotyy(x,y,xb,y_dir_deriv);grid on;
xlabel('time (S)'); ylabel('Voltage (V)');
title(['PSPL Pulser',', ',filthru]);

displayFreq=19.9e9; c=1;
while ( (c<=npts) & (fx(c)<displayFreq) ),
   c=c+1;
end
c=c-1;

f2=fa(1:b+1);
f2(1)=f2(1)/2;
f3=fb(1:b+1);
f3(1)=f3(1)/2;
```

```
  f4=fc_smooth(1:b+1);
  f4(1)=f4(1)/2;


J=810; % J=200;
  figure;
 plot(fx(1:c),20*log10(abs(f2(1:c))),fx(1:c),20*log10(abs(f3(1:c))));grid
on;
  xlabel('frequency (Hz)'); ylabel('(dB)');
  title(['FFTs of PSPL data :',filthru]);
  legend('thru','direct',0);

  displayFreq=11.9e9; c=1;
  while ( (c<=npts) & (fx(c)<displayFreq) ),
     c=c+1;
  end
  c=c 1;


  figure;plot(fx(1:c),20*log10(abs(f4(1:c))));grid on;
  xlabel('frequency (Hz)'); ylabel('(dB)');
  title(['Transfer function of :',', ',filthru]);




%___________________________________________________________________
%
%
[newfile,newpath] =
uiputfile([path_thru,'animinit.xfr'],'Save file name');

if ((newfile(1)~=0) & (newfile(length(newfile))~=0)),
ascwrite_complex(fx2,fc_new,filthru,'legend','Hz','au',[newpath,newfile]);
end




% Compensateit3.m
% 4/99 N Tesny
% corrects multi-rate pulser data
% reads in already-computed transfer function from file

% Set variables:
FALSE=0;TRUE=~FALSE;
%yes=true;no=false;

filpre = 'g:\uwb\other_subjects\lpDecon\';
%filpre = 'c:/dascw/'
%ItemNumber = 6;
% files to calculate transfer function:
%fildirect='drc2pspl.asc';
%filthru='pspllpLogP.asc'; % lp
%fildirect='drc2pspl.asc';
%filthru='psplT220SS.asc'; % t2
```

```
% ask user name of file for thru:
pname=filpre;
[filthru,pname] = uigetfile([pname,'*.asc'],'Select THRU
file')
if ((filthru(1)~=0) & (filthru(length(filthru))~=0)),
   path_thru=pname;
else
   path_thru=filpre;
end

% enter attenuation for THRU file:
OKflag=FALSE;
while OKflag==FALSE,
   prompt  = {'Enter dB attenuation:','Enter ratio attenua-
tion:'};
   title1   = 'Input Attenuation for THRU file';
   lines= 1;
   def     = {'0','2'};
   answer  = inputdlg(prompt,title1,lines,def)

   f=cell2struct(answer(1),'d',1);
   a1s=f.d;
   a1=str2num(a1s);
   f=cell2struct(answer(2),'d',1);
   a2s=f.d;
   a2=str2num(a2s);

   OKflag=TRUE;
   if (isempty(a1)),
      h=errordlg([a1s,' is not a number'],'Bum Data Entry');
      OKflag=FALSE;
      waitfor(h);
   elseif (isempty(a2)),
      h=errordlg([a2s,' is not a number'],'Bum Data Entry');
      OKflag=FALSE;
      waitfor(h);
   end
end
if a2==0,a2=1;end;
thru_attenuation=a1+20*log10(a2)


% ask user name of file for direct:
pname=filpre;
[fildirect,pname] = uigetfile([pname,'*.asc'],'Select DIRECT
file')
if ((fildirect(1)~=0) & (fildirect(length(fildirect))~=0)),
   path_direct=pname;
else
   path_direct=filpre;
end

% enter attenuation for DIRECT file:
OKflag=FALSE;
while OKflag==FALSE,
   prompt  = {'Enter dB attenuation:','Enter ratio attenua-
tion:'};
   title1   = 'Input Attenuation for DIRECT file';
   lines= 1;
   def     = {'20','2'};
```

```
    answer   = inputdlg(prompt,title1,lines,def)

    f=cell2struct(answer(1),'d',1);
    a1s=f.d;
    a1=str2num(a1s);
    f=cell2struct(answer(2),'d',1);
    a2s=f.d;
    a2=str2num(a2s);

    OKflag=TRUE;
    if (isempty(a1)),
       h=errordlg([a1s,' is not a number'],'Bum Data Entry');
       OKflag=FALSE;
       waitfor(h);
    elseif (isempty(a2)),
       h=errordlg([a2s,' is not a number'],'Bum Data Entry');
       OKflag=FALSE;
       waitfor(h);
    end
end
if a2==0,a2=1;end;
direct_attenuation=a1+20*log10(a2)

% ask user name of file for TRANSFER FUNCTION:
pname=filpre;
[filtransfunc,pname] = uigetfile([pname,'*.xfr'],'Select
file to use for TRANSFER FUNCTION')
if ((filtransfunc(1)~=0) &
(filtransfunc(length(filtransfunc))~=0))
   path_transfunc=pname;
else
   path_transfunc=filpre;
end


% Read in files:
filname = strcat(path_thru,filthru);
[x,y] = ascread(filname);
y = y * 10^(thru_attenuation/20); % scale for attenuation

% plot thru:
[pk,fw,ris,x,yjunk] = stats(x,abs(y));
figure;
plot(x,y);grid on;
title('Captured Signal');
v=axis;
a=v(1) + (v(2)-v(1))*.55;
b=v(3) + (v(4)-v(3))*.95;
text(a,b,['rise time (blue): ',num2str(ris/1e-12),' pS']);

% read in transfer function:
filnamt = strcat(path_transfunc,filtransfunc);
[x_tf,y_real,y_imag] = ascread_complex(filnamt);
y_tf=y_real+j*y_imag;
% plot tf:
figure;plot(x_tf,20*log10(abs(y_tf)));grid on;
title('Transfer Function');

% Read in direct file:
```

```
filnameDirect = strcat(path_direct,fildirect);
[x_dir,y_dir] = ascread(filnameDirect);
y_dir = y_dir * 10^(direct_attenuation/20); % scale for
attenuation

y_dir_fft=fft(y_dir);


npts2_d=arraylen(y_dir_fft);
b=fix(npts2_d/2);
npts_d=b+1;

[I,J]=size(x_dir);
a=x_dir(2)-x_dir(1);
fx_d=(0:npts_d-1)/((J*a)); % half freq data
% fx2_d=(0:npts2_d-1)/((J*a)); % full freq data



% plot direct and its fft:
figure,%suptitle(filthru);
subplot(2,1,1),
plot(x_dir,y_dir);
grid on; xlabel('time (S)'); ylabel('Voltage (V)');
title('Response of Direct');

Idir=1;
displayfreq=10e9;
displayfreq=min(displayfreq,fx_d(length(fx_d)));
while fx_d(Idir)<displayfreq,    Idir=Idir+1;  end
Idir=Idir-1;

subplot(2,1,2),
plot(fx_d(1:Idir),20*log10(abs(y_dir_fft(1:Idir))))
grid on; xlabel('frequency (Hz)'); ylabel('(dB)');
title('FFTs');




% Check to see if we need to interpolate:
% ie, if sampling rates are different:
% take fft:
fa=fft(y);
npts2=arraylen(fa);
b=fix(npts2/2);
npts=b+1;

[I,J]=size(x);
a=x(2)-x(1);
fx=(0:npts-1)/((J*a)); % half freq data
fx2=(0:npts2-1)/((J*a)); % full freq data

if abs((x_tf(2)-fx2(2)))/fx2(2) > 9e-7,
   h=warndlg(['Sampling rates are different.  Press OK to
interpolate the transfer function'],'Note');
   waitfor(h);
```

```
   % break the fft and tf in half:
   npts_full = length(fa);
   npts_half = ceil(npts_full/2+.5);
   fa_half=fa(1:npts_half);
   fa_half(1)=fa_half(1)/2; % divide dc value in half

   npts_full_tf = length(y_tf);
   npts_half_tf = ceil(npts_full_tf/2+.5);
   tf_half = y_tf(1:npts_half_tf);
   tf_half(1) = tf_half(1)/2; % divide dc value in half
   x_tf_half = x_tf(1:npts_half_tf);

   % do interpolation of tf:
   %method='linear';
   method='nearest';
   y_tf_interp = interp1(x_tf_half, tf_half, fx, method);

   % now reconstruct inturpolated wf to full length:
   % remember that mirrored side is complex conjugate!
   yTfInterpFull=ones(1,npts_full);
   yTfInterpFull(1:npts_half)=y_tf_interp;
   for i=2:npts_half-1,
      yTfInterpFull(npts_full+2-i) = conj(yTfInterpFull(i));
   end
   yTfInterpFull(1) = yTfInterpFull(1) * 2; % mult dc value
by 2
else
   npts_full = length(fa);
   npts_half = ceil(npts_full/2+.5);
   yTfInterpFull = y_tf;
end

% Now apply transfer function to fft:
fb=fa./yTfInterpFull;

% Filter it out:
filterflag=FALSE;
fpHi=3e9;
fsHi=6e9;
fpLo=200e6;
fsLo=50e6;
delHi=-80;delLo=-20;
fpDi=10e9;
fsDi=20e9; delDi=-40;

% begin filtering
loop___________________________________________________
while filterflag==FALSE,
   OKflag=FALSE;
   while OKflag==FALSE,
      prompt  = {'Enter F-passband in GHz:','Enter F-
stopband in GHz:','Enter stopband loss in dB (delta);'};
      title1   = 'HIGH Frequency Filtering';
      lines= 1;
      def     = {num2str(fpHi/1e9),num2str(fsHi/
1e9),num2str(delHi)};
      answer  = inputdlg(prompt,title1,lines,def)

      f=cell2struct(answer(1),'d',1);
      a1s=f.d;
```

```
        fpHi=str2num(a1s);
        f=cell2struct(answer(2),'d',1);
        a2s=f.d;
        fsHi=str2num(a2s);
        f=cell2struct(answer(3),'d',1);
        a3s=f.d;
        delHi=str2num(a3s);

        OKflag=TRUE;
        if (isempty(fpHi)),
           h=errordlg([a1s,' is not a number'],'Bum Data
Entry');
           OKflag=FALSE;
           waitfor(h);
        elseif (isempty(fsHi)),
           h=errordlg([a2s,' is not a number'],'Bum Data
Entry');
           OKflag=FALSE;
           waitfor(h);
        elseif (isempty(delHi)),
           h=errordlg([a3s,' is not a number'],'Bum Data
Entry');
           OKflag=FALSE;
           waitfor(h);
        end
     end
     fpHi=fpHi*1e9;
     fsHi=fsHi*1e9;

     % Let user enter low freq numbers:
     OKflag=FALSE;
     while OKflag==FALSE,
        prompt  = {'Enter F-stopband in MHz:','Enter F-pass-
band in MHz:','Enter stopband loss in dB (delta);'};
        title1   = 'LOW Frequency Filtering';
        lines= 1;
        def     = {num2str(fsLo/1e6),num2str(fpLo/
1e6),num2str(delLo)};
        answer  = inputdlg(prompt,title1,lines,def)

        f=cell2struct(answer(1),'d',1);
        a1s=f.d;
        fsLo=str2num(a1s);
        f=cell2struct(answer(2),'d',1);
        a2s=f.d;
        fpLo=str2num(a2s);
        f=cell2struct(answer(3),'d',1);
        a3s=f.d;
        delLo=str2num(a3s);

        OKflag=TRUE;
        if (isempty(fpHi)),
           h=errordlg([a1s,' is not a number'],'Bum Data
Entry');
           OKflag=FALSE;
           waitfor(h);
        elseif (isempty(fsHi)),
```

```
            h=errordlg([a2s,' is not a number'],'Bum Data
Entry');
            OKflag=FALSE;
            waitfor(h);
         elseif (isempty(delHi)),
            h=errordlg([a3s,' is not a number'],'Bum Data
Entry');
            OKflag=FALSE;
            waitfor(h);
         end
   end
   fpLo=fpLo*1e6;
   fsLo=fsLo*1e6;

   % let user enter limits for direct waveform filter:
   OKflag=FALSE;
   while OKflag==FALSE,
      prompt  = {'Enter F-passband in GHz:','Enter F-
stopband in GHz:','Enter stopband loss in dB (delta);'};
      title1    = 'HIGH Frequency Filtering For DIRECT Wave-
form';
      lines= 1;
      def      = {num2str(fpDi/1e9),num2str(fsDi/
1e9),num2str(delDi)};
      answer  = inputdlg(prompt,title1,lines,def)

      f=cell2struct(answer(1),'d',1);
      a1s=f.d;
      fpDi=str2num(a1s);
      f=cell2struct(answer(2),'d',1);
      a2s=f.d;
      fsDi=str2num(a2s);
      f=cell2struct(answer(3),'d',1);
      a3s=f.d;
      delDi=str2num(a3s);

      OKflag=TRUE;
      if (isempty(fpHi)),
         h=errordlg([a1s,' is not a number'],'Bum Data
Entry');
         OKflag=FALSE;
         waitfor(h);
      elseif (isempty(fsHi)),
         h=errordlg([a2s,' is not a number'],'Bum Data
Entry');
         OKflag=FALSE;
         waitfor(h);
      elseif (isempty(delHi)),
         h=errordlg([a3s,' is not a number'],'Bum Data
Entry');
         OKflag=FALSE;
         waitfor(h);
      end
   end
   fpDi=fpDi*1e9;
   fsDi=fsDi*1e9;


   % Generate filter from user specs:
   for i=1:length(fx),
      if fx(i)<=fsLo,
```

```
            filt1(i)=delLo;
         elseif fsLo<fx(i) & fx(i)<fpLo,
            filt1(i) = delLo * (fpLo-fx(i)) / (fpLo-fsLo);
         elseif fpLo<=fx(i) & fx(i)<=fpHi,
            filt1(i)=0;
         elseif fpHi<fx(i) & fx(i)<fsHi,
            filt1(i) = delHi * (fpHi-fx(i)) / (fpHi-fsHi);
         elseif fx(i)>=fsHi,
            filt1(i) = delHi;
         end
         filt1(i) = 10 ^ (filt1(i)/20); % convert from dB to
number
      end


      % now reconstruct filter to full length:
      filt_full=zeros(1,npts_full);
      filt_full(1:npts_half)=filt1;
      for i=2:npts_half-1,
         filt_full(npts_full+2-i) = filt_full(i);
      end

      figure;plot(20*log10(filt_full));

      % Scale with filter:
      fbFilt = fb .* filt_full;

      % Do ifft:
      y_new = ifft(fbFilt);

      % check waveform for complex data:
      n=min(40,length(y_new));
      warningFlag=FALSE;
      for i=1:n,
         if abs(imag(y_new(i))) > 9e-7,
            warningFlag=TRUE;
         end
      end
      if (warningFlag==TRUE),
         h=warndlg(['Complex data is present in the time domain
waveform!'],'Warning');
         waitfor(h);
      end

      figure;plot(x,real(y_new));

      % check for sample rates being different:
      if abs((x_dir(2)-x(2)))/x(2) > 9e-7,
         h=warndlg(['Sample rate of Direct is different from
Thru.  Press OK to continue'],'Note');
         waitfor(h);
      end

      % Take derivative of direct:
      dt=x_dir(2)-x_dir(1); dy(1)=0;
      for i=2:length(y_dir),
         dy(i)=(y_dir(i)-y_dir(i-1));
      end
      % scale direct deriv:
      y_dir_deriv = dy * max(abs(y_dir)) / max(abs(dy));
```

```
dirDerivFFT = fft(y_dir_deriv);

npts2_d=arraylen(dirDerivFFT);
b=fix(npts2_d/2);
npts_d=b+1;

[I,J]=size(x_dir);
a=x_dir(2)-x_dir(1);
fx_d=(0:npts_d-1)/((J*a)); % half freq data
fx2_d=(0:npts2_d-1)/((J*a)); % full freq data


% Generate filter for direct from user specs:
for i=1:length(fx_d),
   if fx_d(i)<=fpDi,
      filtD(i) = 0;
   elseif fpDi<fx_d(i) & fx_d(i)<fsDi,
      filtD(i) = delDi * (fpDi-fx_d(i)) / (fpDi-fsDi);
   elseif fx_d(i)>=fsDi,
      filtD(i) = delDi;
   end
   filtD(i) = 10 ^ (filtD(i)/20); % convert from dB to
number
end


% now reconstruct filter to full length:
filt_fullD=zeros(1,npts2_d);
filt_fullD(1:npts_d)=filtD;
for i=2:npts_d-1,
   filt_fullD(npts2_d+2-i) = filt_fullD(i);
end


% Scale with filter:
dirDerivFFT_filt = dirDerivFFT .* filt_fullD;

% Do ifft:
y_dir_deriv_new = ifft(dirDerivFFT_filt);
%figure;plot(x,real(y_dir_new));

figure;plot(x_dir,y_dir,
x_dir,real(y_dir_deriv_new));grid on;
title('Filtered direct waveform');
xlabel('time (S)');ylabel('V');
legend('direct','direct-deriv');


% Plot corrected and direct:
figure;plot(x,y_new,x_dir,y_dir_deriv_new);
grid on;xlabel('time (S)');ylabel('V');
title('Results');legend('corrected','direct deriv');

figure;plot(fx, 20*log10(abs(fbFilt(1:npts_half))),...
   fx_d, 20*log10(abs(dirDerivFFT_filt(1:npts_d))));
grid on;xlabel('freq (Hz)');ylabel('V/Hz');
title('Results');legend('corrected','direct deriv');

displayFreq=14.9e9; b=1;
```

```
    while ( (b<=npts_half) & (fx(b)<displayFreq) ),
       b=b+1;
    end
    b=b-1;

    displayFreq=14.9e9; bd=1;
    while ( (bd<=npts_d) & (fx_d(bd)<displayFreq) ),
       bd=bd+1;
    end
    bd=bd-1;

    figure;plot(fx(1:b),20*log10(abs(filt1(1:b))),'r-',...
       fx(1:b),20*log10(abs(fb(1:b))),'k-',...
       fx(1:b),20*log10(abs(fbFilt(1:b))),'b-',...
       fx_d(1:bd),20*log10(abs(dirDerivFFT_filt(1:bd))),'g-
');
    grid on;xlabel('freq (Hz)');ylabel('V/Hz');
   title('Results');legend('filter','unfiltered','filtered','direct
deriv');

    filterflag=TRUE;
    button = questdlg('Are filter limits OK?',...
       'Continue Operation','Yes','No','Help','No')
    if strcmp(button,'Yes')
       disp('OK')
    elseif strcmp(button,'No')
       disp('Redo filter operation')
       filterflag=FALSE;
       close 4;close 5;close 6;close 7;close 8;close 9;
    elseif strcmp(button,'Help')
       disp('Sorry, no help available')
    end

end; % end of loop for filterflag
```

```
[pk1,fw1,ris1,xjunk,yjunk] = stats(x,real(y_new));
rt1=num2str(round(ris1/1e-12));
[pk2,fw2,ris2,xjunk,yjunk] = stats(x,y);
rt2=num2str(round(ris2/1e-12));
[pk3,fw3,ris3,xjunk,yjunk] =
stats(x_dir,real(y_dir_deriv_new)); rt3=num2str(round(ris3/
1e-12));


% normalize and plot on same graph:
figure,suptitle(filthru);
subplot(2,1,1),
[xAlign1,y_newAlign]=align_peak(x,y_new);
[xAlign2,yAlign]=align_peak(x,y);
[xAlign3,dyAlign]=align_peak(x_dir,y_dir_deriv_new);

timbeg=-2e-9;timend=2e-9;
if strncmpi(filthru,'av',2),
   timbeg=-9.9e-9;timend=20e-9; %if avtek, display longer
window
end
if strncmpi(filthru,'hhlp',4),
```

```
   timbeg=-9.9e-9;timend=20e-9; %if lp, display longer
window
end
timbeg=-20e-9;timend=20e-9;

[xAlign1,y_newAlign]=prepst(xAlign1,y_newAlign,timbeg,timend);
[xAlign2,yAlign]=prepst(xAlign2,yAlign,timbeg,timend);
[xAlign3,dyAlign]=prepst(xAlign3,dyAlign,timbeg,timend);

if abs(max(yAlign))<abs(min(yAlign)),
   yAlign=-yAlign;
end

plot(xAlign1,y_newAlign/max(y_newAlign),'g',xAlign2,yAlign/
max(yAlign),...
   'b',xAlign3,dyAlign/max(dyAlign),'r');
grid on; xlabel('time (S)'); ylabel('Voltage (V)');
%title('Response of Pulser');
legend(['comp, rt=',rt1,' ps'],['meas, rt=',rt2,' ps'],['d/
dt, rt=',rt3,' ps']);
I=1;
displayfreq=10e9;
displayfreq=min(displayfreq,fx(length(fx)));
while fx(I)<displayfreq,    I=I+1;  end
I=I-1;

Idir=1;
displayfreq=10e9;
displayfreq=min(displayfreq,fx_d(length(fx_d)));
while fx_d(Idir)<displayfreq,    Idir=Idir+1;  end
Idir=Idir-1;

subplot(2,1,2),
plot(fx(1:I),20*log10(abs(fa(1:I))),...
   fx(1:I),20*log10(abs(fbFilt(1:I))),...
   fx_d(1:Idir),20*log10(abs(dirDerivFFT_filt(1:Idir))) );

grid on; xlabel('frequency (Hz)'); ylabel('(dB)');
title('FFTs');
legend('meas','comp','d/dt',0);
```

Appendix B. Examples of Compensation

This appendix shows examples of the original data and the intermediate results as the data are processed according to the flow charts discussed in figures 6 and 7. Figures B-1 through B-4 show the direct 4015 signal recorded on the scope as well as that recorded through the T3 antenna pair. Figures B-5 through B-8 show the fast Fourier transforms of these recorded signals. Figures B-9 through B-12 show the calculated transfer functions of the 4015 through the four antennas, and figures B-13 through B-21 show the reconstructed signals. The reconstructions are plotted over the original signal for comparison, and these compare well. The complete matrix of sources used during the experiments is shown in table B-1.

Table B-1. Antennas and sources studied.

	Antennas			
Source	T3	T2	LP	EMCO
PSPL 4015C	X	X	X	X
HH2	X	X	X	X
Avtech	X	X	X	X

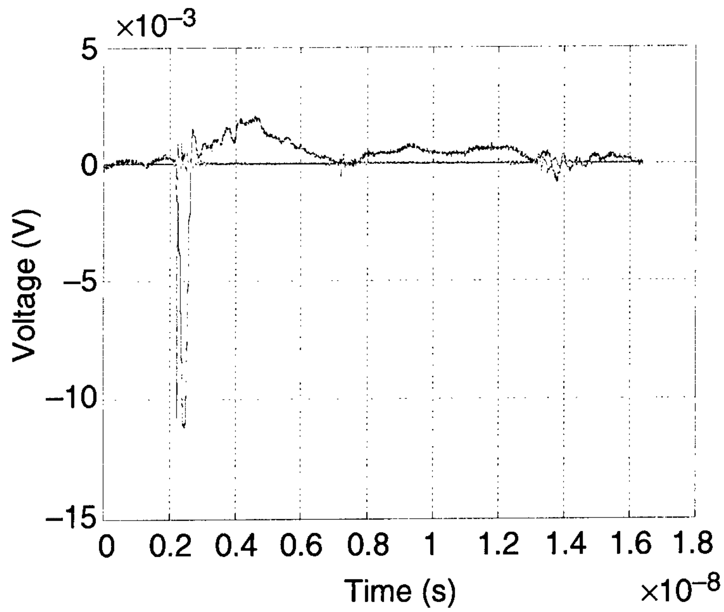


Figure B-1. Signals from PSPL source and T2 antenna pair.

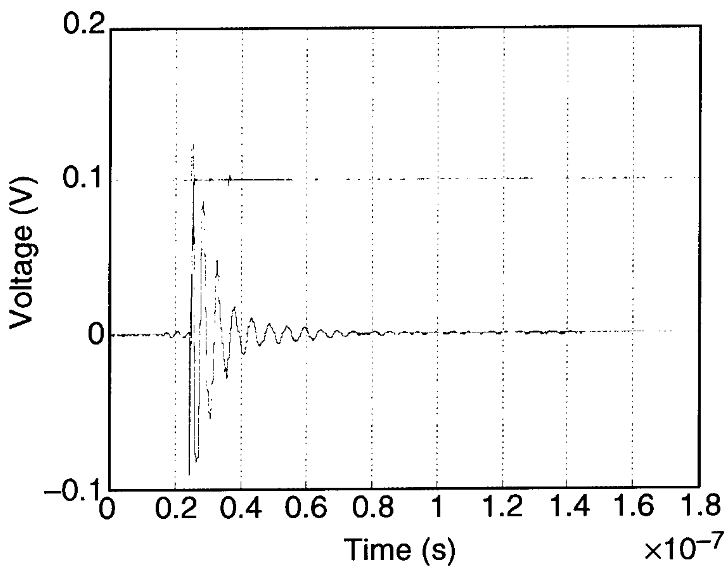


Figure B-2. Signals from PSPL source and EMCO antenna pair.

Figure B-3. Signals from PSPL source and log periodic antenna pair.

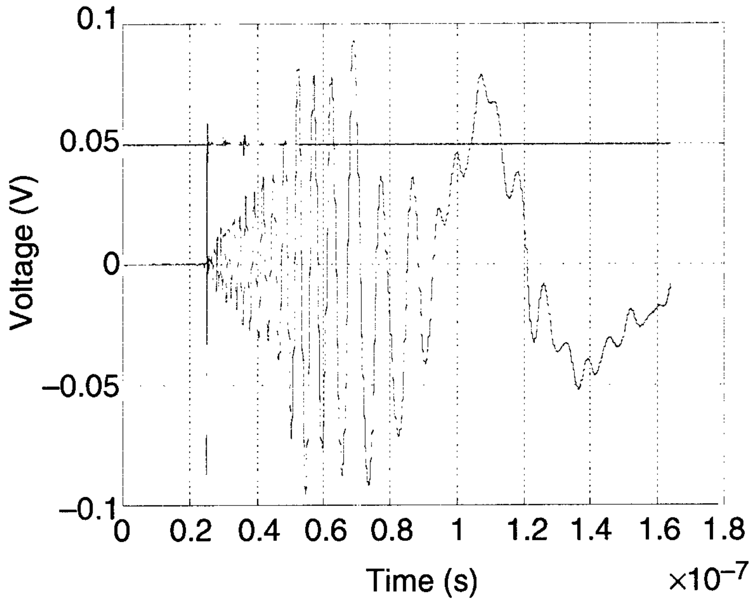


Figure B-4. Signals from PSPL source and T3 antenna pair.

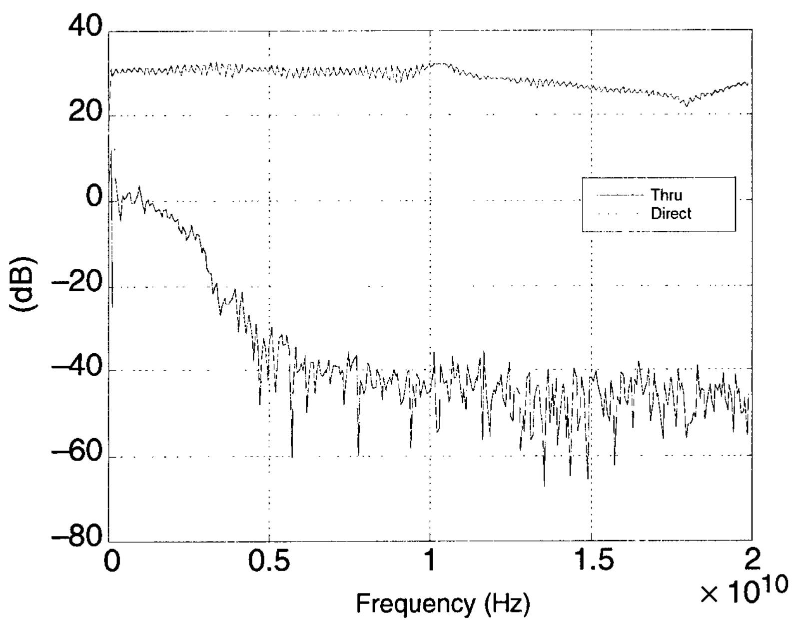


Figure B-5. FFTs of signals from PSPL source and T2 antenna pair.

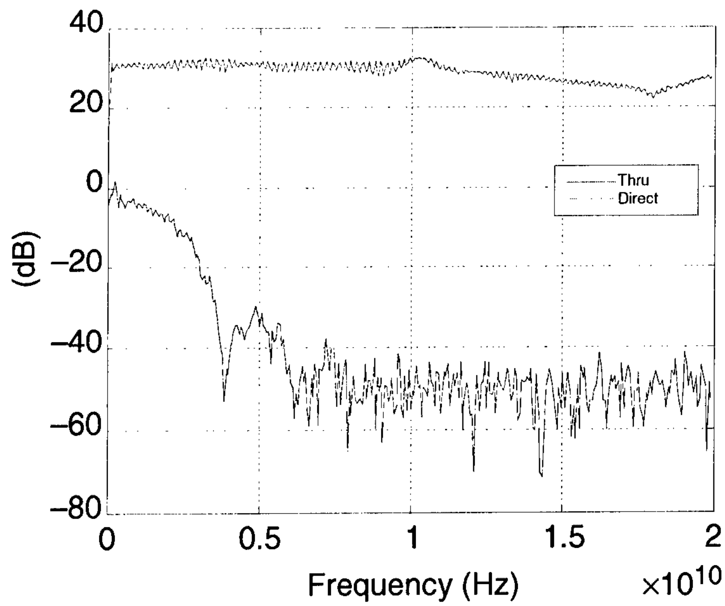


Figure B-6. FFTs of signals from PSPL source and EMCO antenna pair.

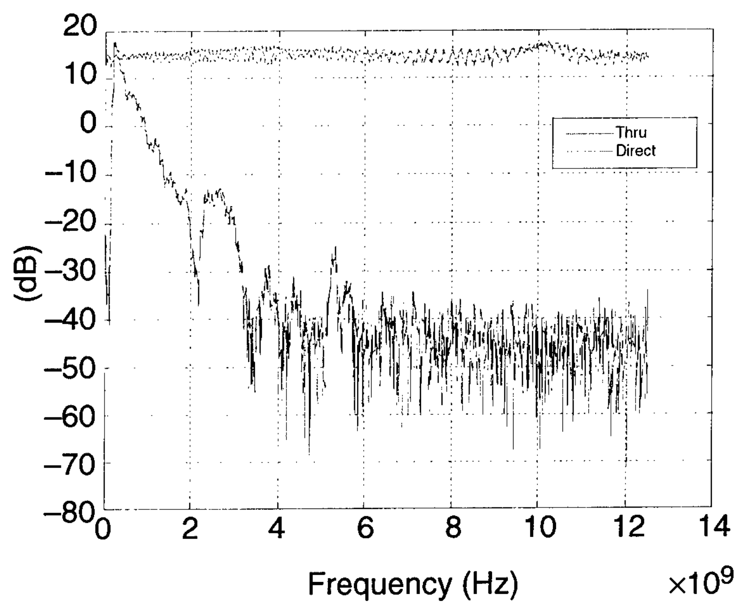


Figure B-7. FFTs of signals from PSPL source and log periodic antenna pair.

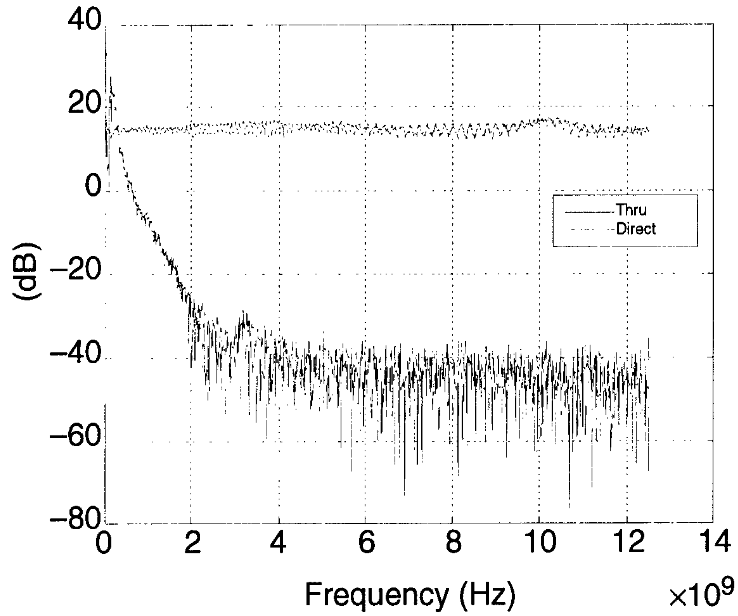


Figure B-8. FFTs of signals from PSPL source and T3 antenna pair.

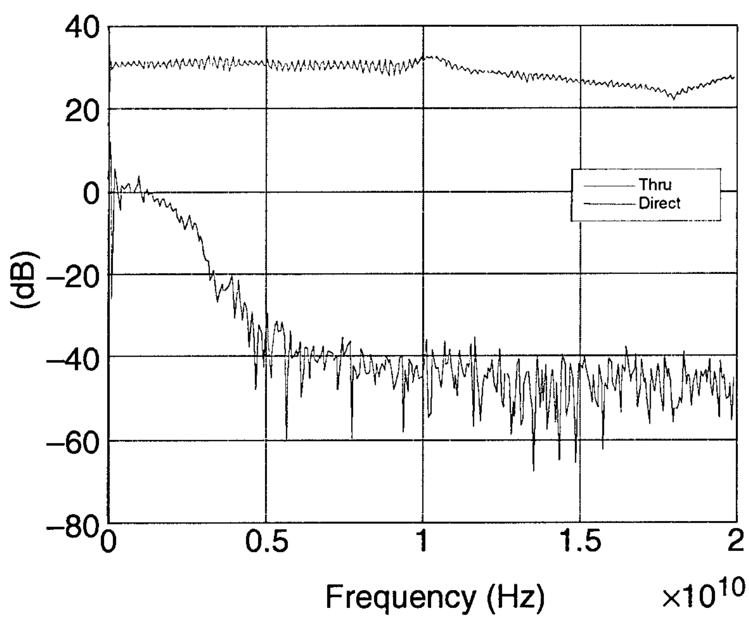


Figure B-9. Transfer function of T2 antenna pair.

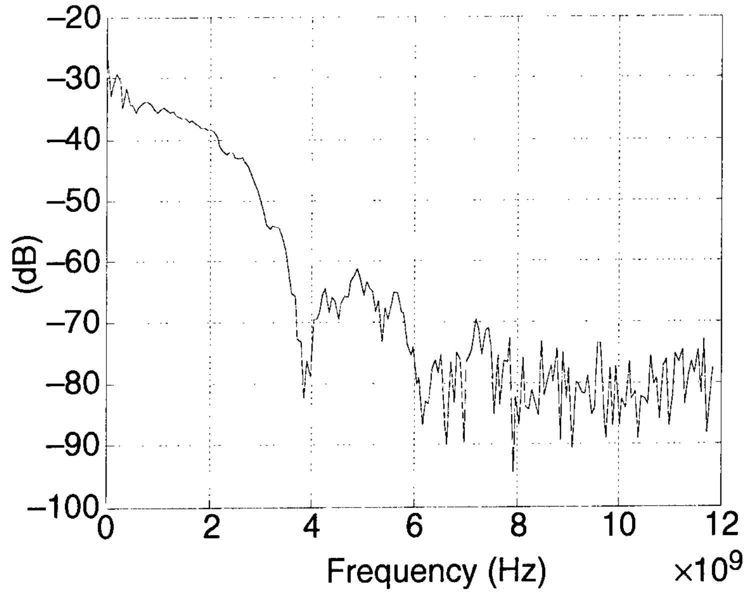


Figure B-10. Transfer function of EMCO antenna pair.

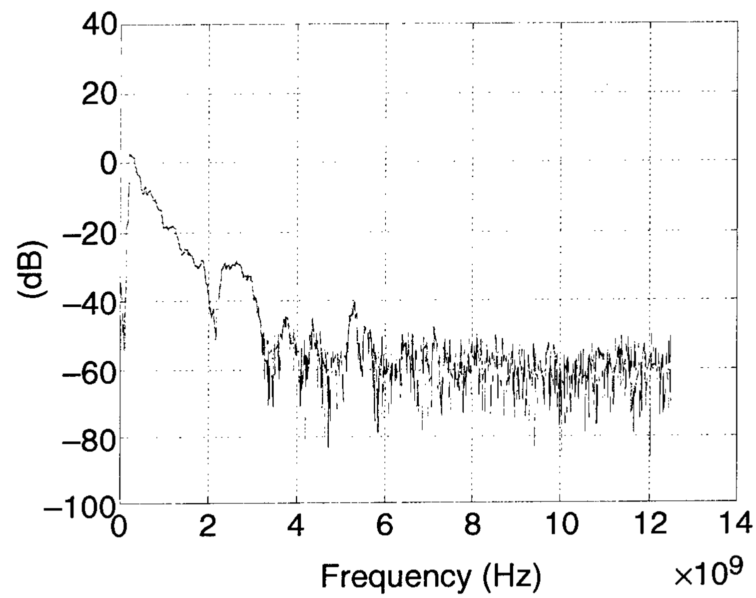


Figure B-11. Transfer function of log periodic antenna pair.

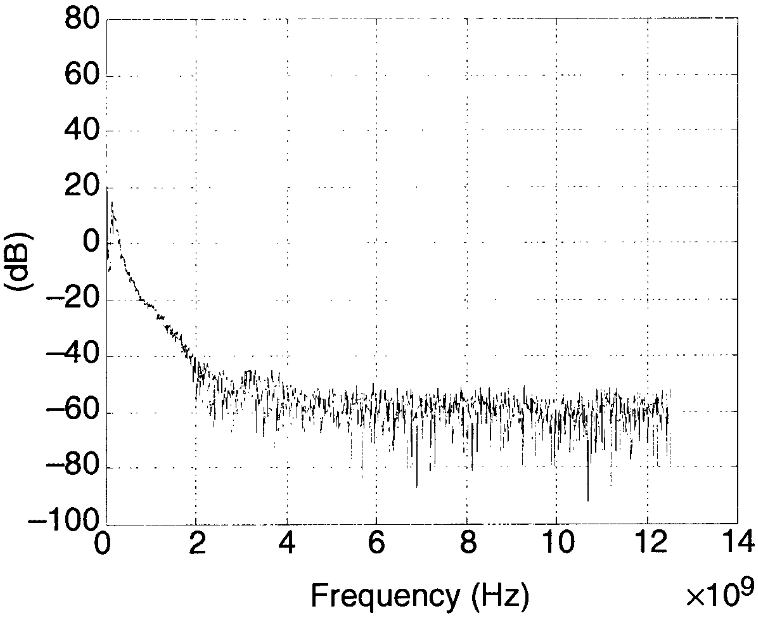


Figure B-12. Transfer function of T3 antenna pair.

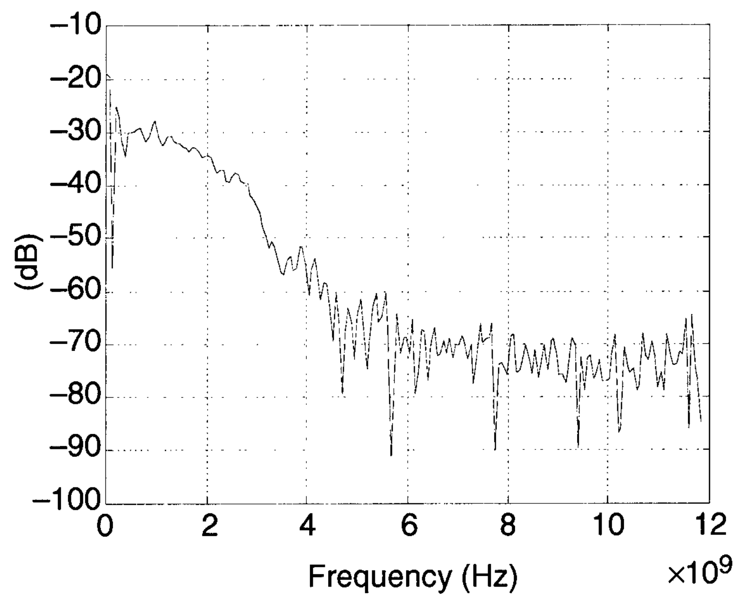


Figure B-13. Reconstruction of signal from HH2 source through log periodic antenna pair.

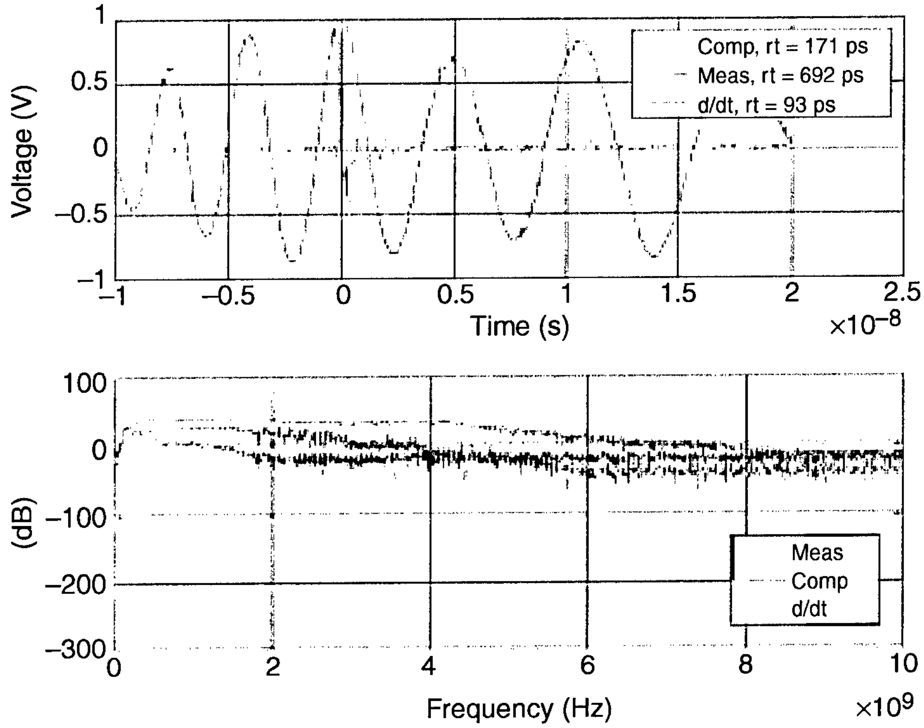


Figure B-14. Reconstruction of signal from Avtech source through LP antenna pair.

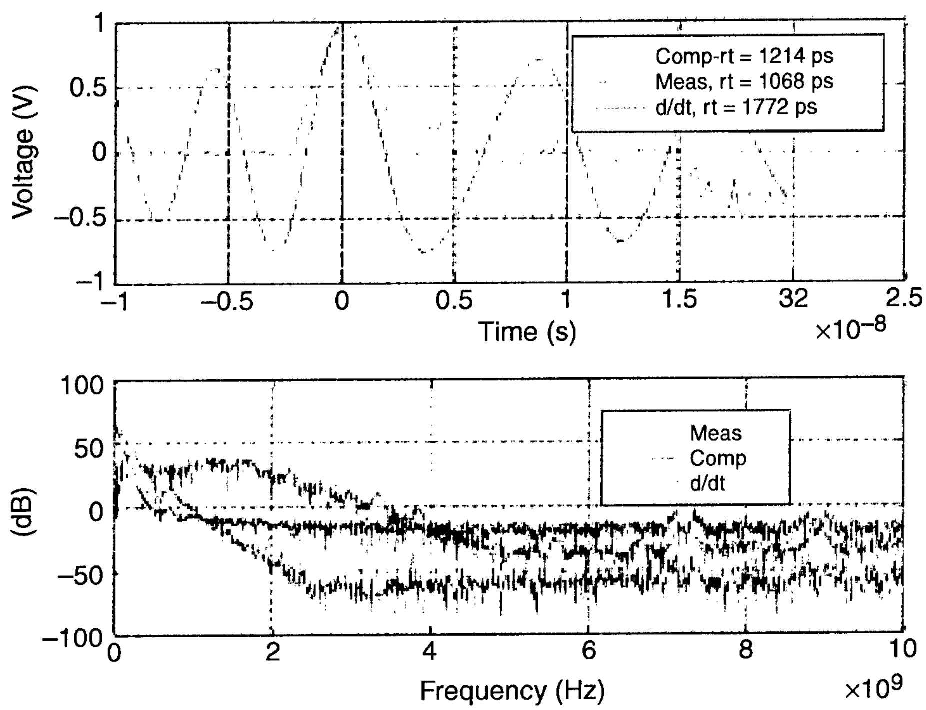


Figure B-15. Reconstruction of signal from HH2 source through T3 antenna pair.

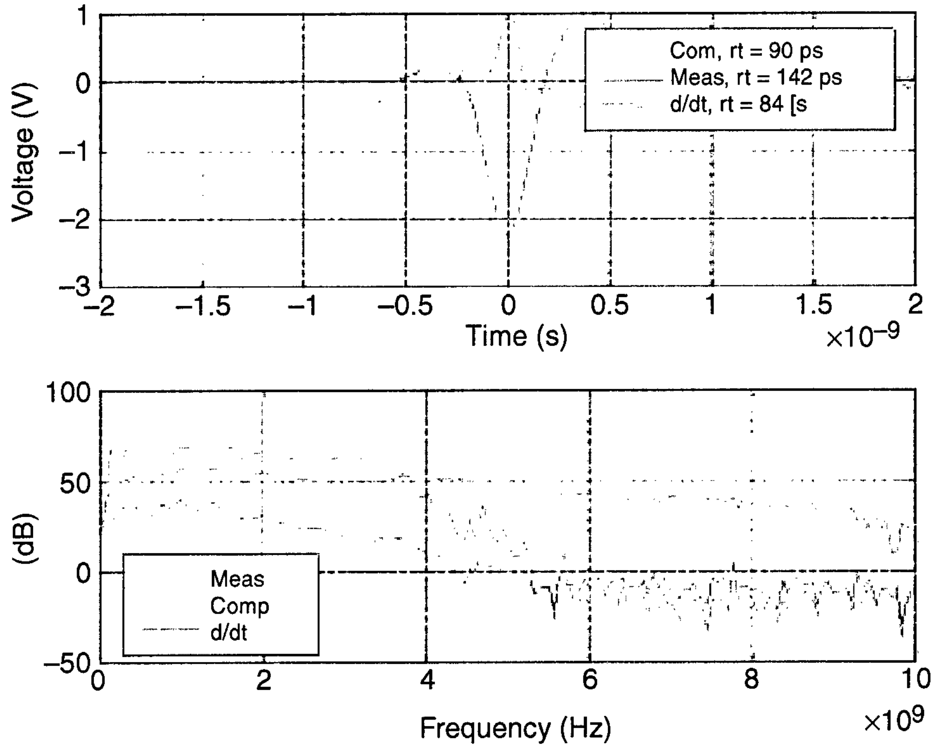


Figure A-16. Reconstruction of signal from Avtech source through T3 antenna pair.

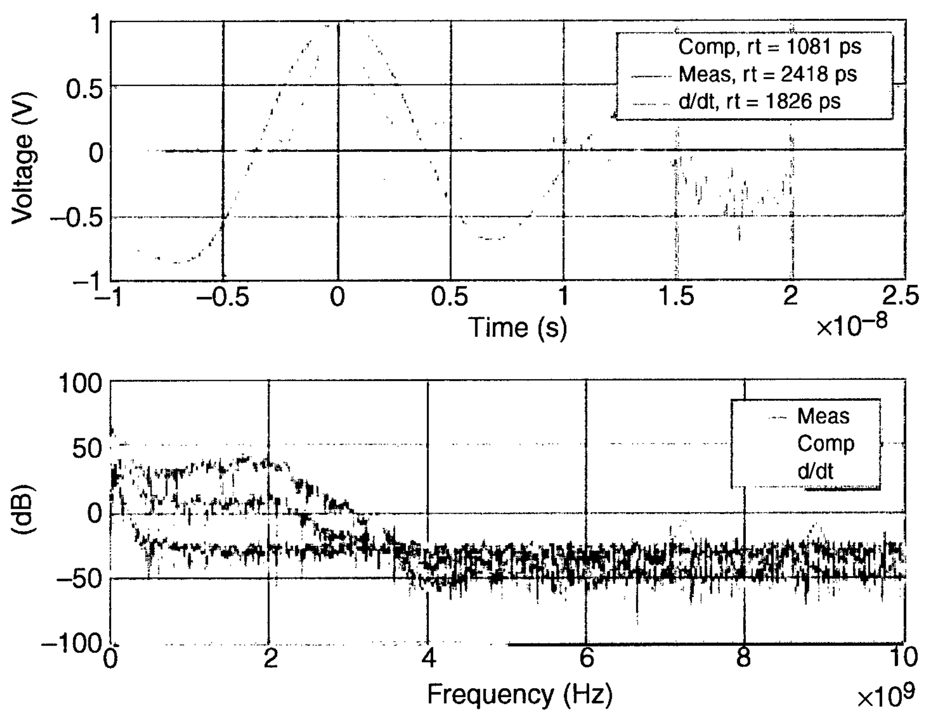


Figure B-17. Jitter in Avtech source.

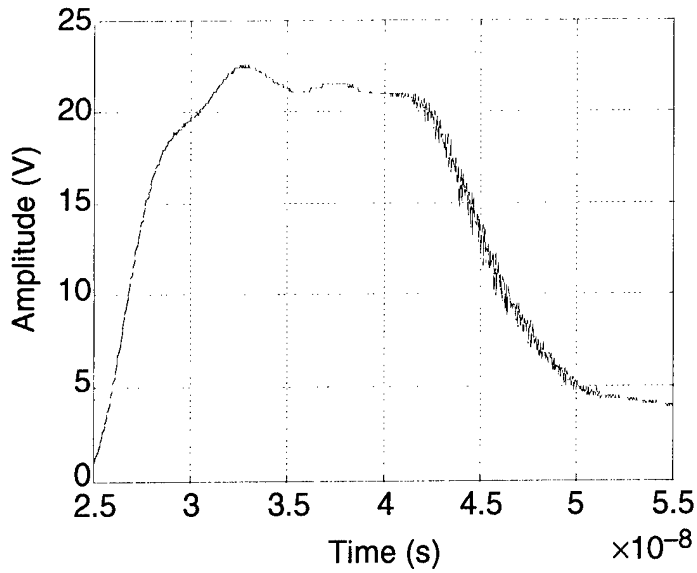


Figure B-18. Multi-rate reconstruction of signal from HH2 source through T2 antenna pair.

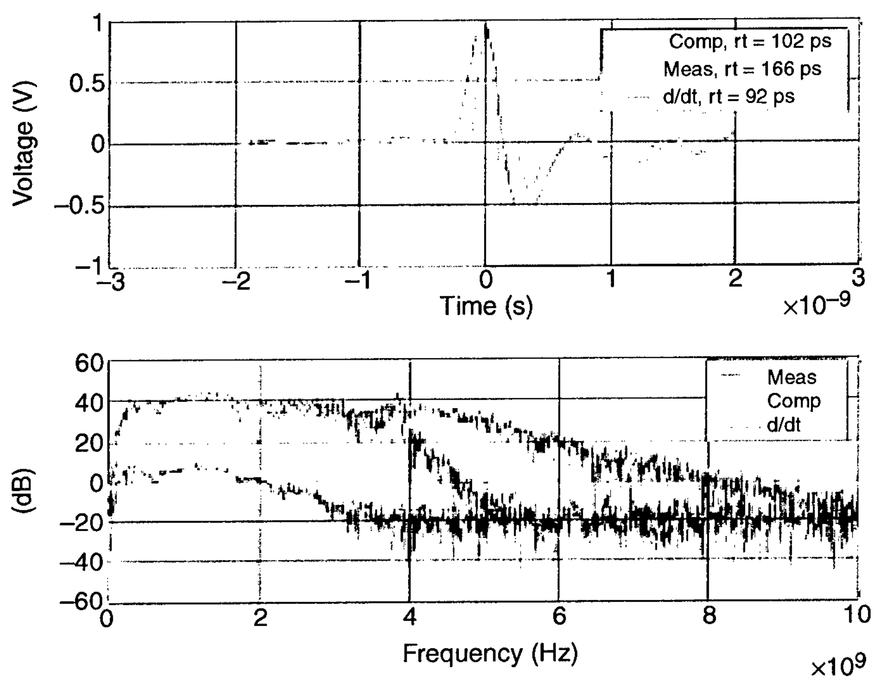


Figure B-19. Multi-rate reconstruction of signal from HH2 source through T3 antenna pair measured with an SCD1000 oscilloscope.

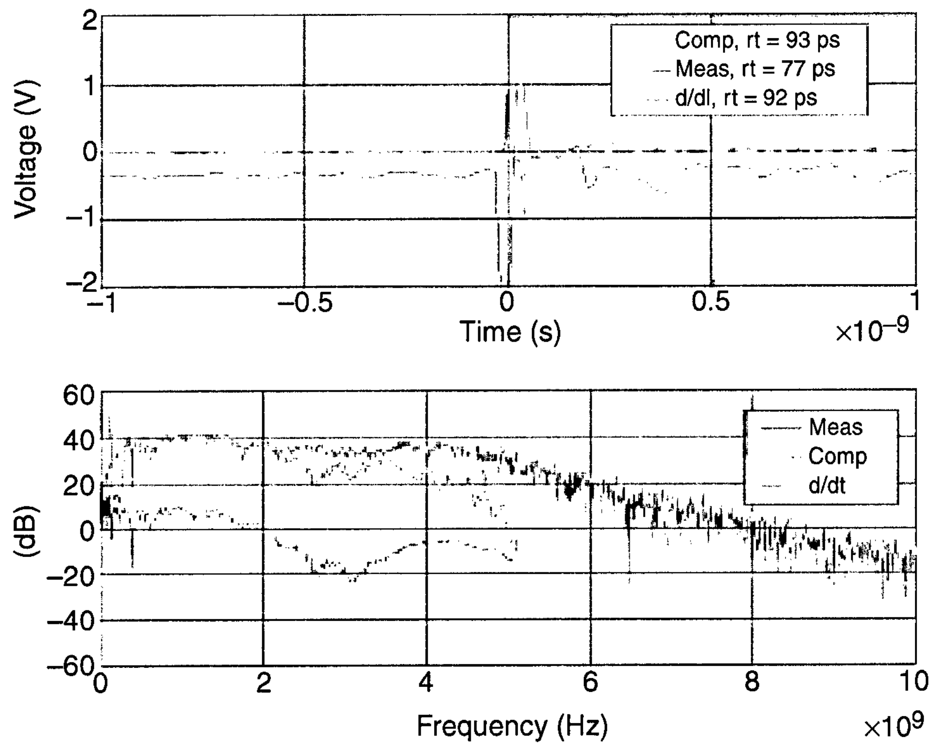


Figure B-20. Multi-rate reconstruction of signal from HH2 source through EMCO antenna pair measured with an SCD1000 oscilloscope.

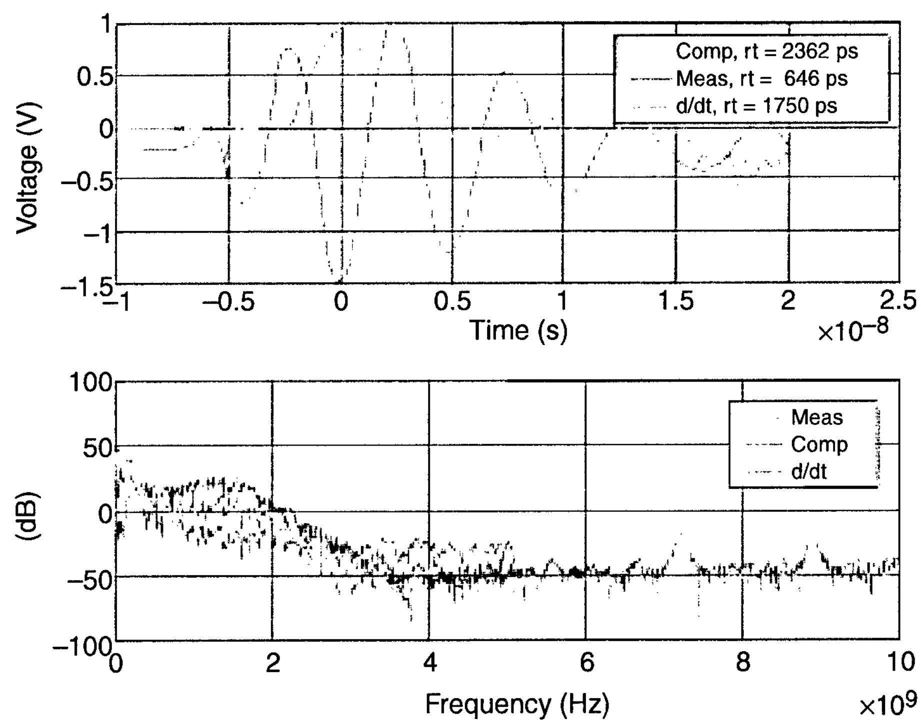
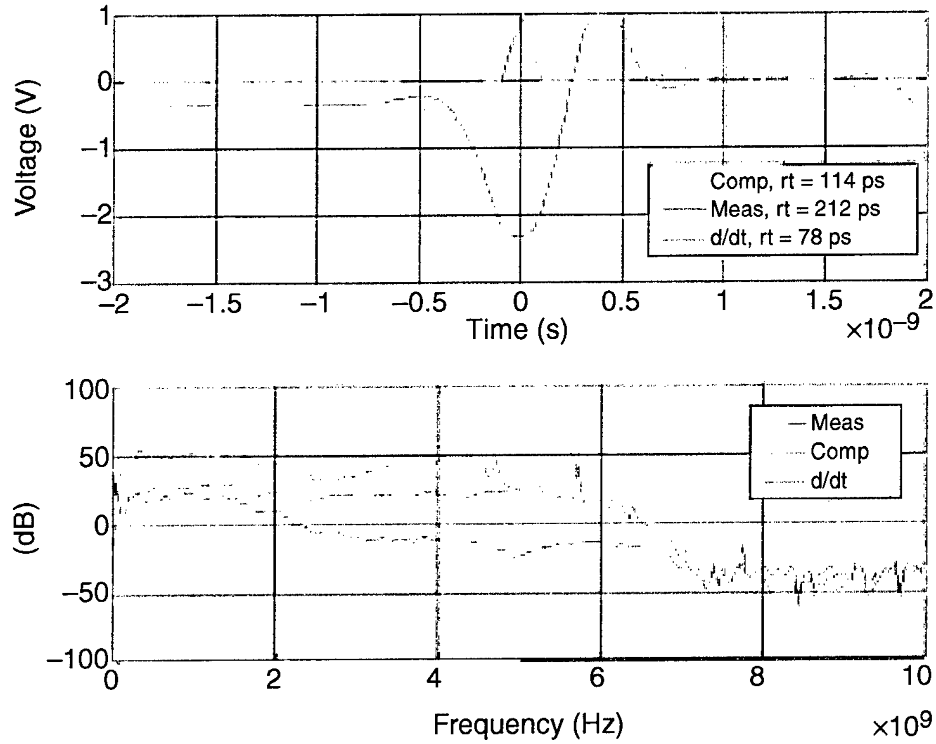


Figure B-21. Multi-rate reconstruction of signal from HH2 source through T3 antenna pair measured with an SCD1000 oscilloscope.



Distribution

Admnstr
Defns Techl Info Ctr
ATTN DTIC-OCP
8725 John J Kingman Rd Ste 0944
FT Belvoir VA 22060-6218

DARPA
ATTN S Welby
3701 N Fairfax Dr
Arlington VA 22203-1714

Ofc of the Secy of Defns
ATTN ODDRE (R&AT)
The Pentagon
Washington DC 20301-3080

AMCOM MRDEC
ATTN AMSMI-RD W C McCorkle
Redstone Arsenal AL 35898-5240

US Army TRADOC
Battle Lab Integration & Techl Dirctrt
ATTN ATCD-BFT Monroe VA 23651-5850

DIRNSA
ATTN D Henkin
9800 Savage Rd
FT Meade MD 20755-6514

US Military Acdmy
Mathematical Sci Ctr of Excellence
ATTN MADN-MATH MAJ M Huber
Thayer Hall
West Point NY 10996-1786

Dir for MANPRINT
Ofc of the Deputy Chief of Staff for Prsnnl
ATTN J Hiller
The Pentagon Rm 2C733
Washington DC 20301-0300

SMC/CZ
A2435 Vela Way Ste 1613
El Segundo CA 90245-5500

TECOM
ATTN AMSTE-CL
Aberdeen Proving Ground MD 21005-5057

US Army ARDEC
ATTN AMSTA-AR-TD
Bldg 1
Picatinny Arsenal NJ 07806-5000

US Army Info Sys Engrg Cmnd
ATTN AMSEL-IE-TD F Jenia
FT Huachuca AZ 85613-5300

US Army Intllgnc & Info Warfare Dirctrt
ATTN AMSEL-RD-IW B Mak
ATTN AMSEL-RD-IW D Helm
ATTN AMSEL-RD-IW K Leshick
ATTN AMSEL-RD-IW MAJ R Martinsen
ATTN AMSEL-RD-IW T Provencher
Bldg 600
FT Monmouth NJ 07703-5211

US Army Natick RDEC Acting Techl Dir
ATTN SBCN-T P Brandler
Natick MA 01760-5002

US Army Simulation Train & Instrmntn Cmnd
ATTN AMSTI-CG M Macedonia
ATTN J Stahl
12350 Research Parkway
Orlando FL 32826-3726

US Army Tank-Automtv Cmnd RDEC
ATTN AMSTA-TR J Chapin
Warren MI 48397-5000

Nav Surfc Warfare Ctr
ATTN Code B07 J Pennella
17320 Dahlgren Rd Bldg 1470 Rm 1101
Dahlgren VA 22448-5100

Hicks & Assoc Inc
ATTN G Singley III
1710 Goodrich Dr Ste 1300
McLean VA 22102

Palisades Inst for Rsrch Svc Inc
ATTN E Carr
1745 Jefferson Davis Hwy Ste 500
Arlington VA 22202-3402Director

Distribution (cont'd)

US Army Rsrch Ofc
ATTN AMSRL-RO-D JCI Chang
ATTN AMSRL-RO-EN W D Bach
PO Box 12211
Research Triangle Park NC 27709

US Army Rsrch Lab
ATTN AMSRL-CI-AI-R Mail & Records Mgmt
ATTN AMSRL-CI-AP Techl Pub (2 copies)
ATTN AMSRL-CI-LL Techl Lib (2 copies)
ATTN AMSRL-DD J M Miller

US Army Rsrch Lab (cont'd)
ATTN AMSRL-IS-TA B Sadler
ATTN AMSRL-SE-D E Scannell
ATTN AMSRL-SE-DP C Lazard
ATTN AMSRL-SE-DP L Dilks (10 copies)
ATTN AMSRL-SE-DP M Litz (10 copies)
ATTN AMSRL-SE-DP N Tesny (10 copies)
ATTN AMSRL-SE-DP R A Kehs
ATTN AMSRL-SE-RU J Sichina
ATTN AMSRL-SE-RU M Ressler
Adelphi MD 20783-1197

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE July 2001

3. REPORT TYPE AND DATES COVERED Final, FY00

4. TITLE AND SUBTITLE Antenna Transient Compensation

5. FUNDING NUMBERS
DA PR: N/A
PE: 62120A

6. AUTHOR(S) N. Tesny, M. Litz, L. Dilks, and D. Conrad

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
Attn: AMSRL-SE-DP email: ntesny@arl.army.mil
2800 Powder Mill Road
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-2229

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD 20783-1197

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
ARL PR: 1NX1XX
AMS code: 622120.H16

12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

An automated method has been implemented in MATLAB® to compensate for signal dispersion in antenna structures. We have explored postprocessing techniques that involve frequency transforms and deconvolution. The method has been applied to transient signals measured from a variety of different antennas and impulse sources. The technique has proved to be a valuable tool in reconstructing fast transient signals with inexpensive high-gain log-periodic antennas instead of more expensive, high fidelity wideband horns.

14. SUBJECT TERMS UWB, URINT

15. NUMBER OF PAGES 52

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102